DIGITAL ENCODING
FOR
SECURE DATA COMMUNICATIONS

Eduardo Emilio Coquis Rondón

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

DIGITAL ENCODING
FOR
SECURE DATA COMMUNICATIONS

by

Eduardo Emilio Coquis Rondón

September 1976

Thesis Advisor: G. Marmont

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Digital Encoding for Secure Data Communications		5. TYPE OF REPORT & PERIOD COVERED Engineer's Thesis; September 1976
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Eduardo Emilio Coquis Rondón		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		12. REPORT DATE September 1976
		13. NUMBER OF PAGES 124
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Digital Encoding
Cryptography
pseudo-random cipher
data-keyed cipher

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This thesis is concerned with the use of the digital computer to realize cryptography. Three cryptographic systems: simple substitution, pseudo-random cipher (polyalphabetic cipher), and data-keyed cipher, are

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
(Page 1) S/N 0102-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

(20. ABSTRACT Continued)

designed, implemented through computer programming, and evaluated. A suitable cyclic error correcting code is designed to encode these systems for transmission. The code is tested by simulating a noisy channel.

Digital Encoding
for
Secure Data Communications

by

Eduardo Emilio Coquis Rondón
Lieutenant, Peruvian Navy
B.S., Naval Postgraduate School, 1974
M.S., Naval Postgraduate School, 1975

Submitted in partial fulfillment of the
requirements for the degree of

ELECTRICAL ENGINEER

from the

NAVAL POSTGRADUATE SCHOOL
September 1976

ABSTRACT

This thesis is concerned with the use of the digital computer to realize cryptography. Three cryptographic systems: simple substitution, pseudo-random cipher (polyalphabetic cipher), and data-keyed cipher, are designed, implemented through computer programming, and evaluated. A suitable cyclic error correcting code is designed to encode these systems for transmission. The code is tested by simulating a noisy channel.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

I. DEFINITIONS

The following definitions are given to acquaint the reader with some of the terms commonly encountered in the field of cryptography.

Cryptology is the branch of knowledge that deals with the development and use of all forms of secret communication.

Cryptography is the branch of cryptology that deals with secret writing.

Cryptanlaysis is the branch of cryptology that deals with the analysis and solution of cryptographic systems.

A Cipher is a cryptographic system which conceals, in a cryptographic sense, the letters or groups of letters in the message or plaintext.

Enciphering is the operation of concealing a plaintext, and the result is a cipher text, or in general a cryptogram.

Deciphering is the process of discovering the secret meaning of a cipher text.

A key is the variable parameter of a cipher system, prearranged between correspondents, which determines the specific application of a general cipher system being used. The use of keys permits almost endless variations within a given cipher system. In fact, the value of a specific cipher system is based on how hard it is for an "enemy" to break a cryptogram or series of cryptograms, assuming he knows the complete details of the system but

lacks the keys which were used to encipher the cryptograms originally.

A <u>code</u> is a cryptographic system which substitutes symbol groups for words, phrases, or sentences found in the plaintext. It involves the use of a codebook, copies of which are kept by each correspondent.

<u>Encoding</u> is the operation of concealing a message using a code.

<u>Decoding</u> is the process of recovering an encoded message.

A code differs from a cipher because a code deals with plaintext in variable size units, such as words or phrases, while a cipher deals with plaintext in fixed size units, usually a letter at a time.

II. INTRODUCTION

Since there is no way of making data communication links physically secure, particularly if some form of radio transmission is involved, encryption is the only practical method of protecting the transmitted data. In the commercial world and nonmilitary parts of government, there is a growing need for encryption. This need for encryption is not just to satisfy the legal requirements for privacy, but also to protect systems from criminal activities.

At the present time, communication systems seem to be going towards digital means. There are already in use digital systems for data communications as well as for public services such as the telephone system.

The present work was intended to study the possibility of using a digital computer to realize cryptographic systems. Further, this computer can be envisioned as part of a digital communication system, mainly to do cryptography and to implement suitable error correcting codes. The DEC PDP-11/40 minicomputer was used to do this study.

Through this work, three cryptographic systems were designed, ranging from a simple substitution cipher to a data-keyed cipher. On the latter the message itself constituted the key to modify other characters. Very significant results were obtained from it in the sense that it gives rise to a text where its characters were nearly

equiprobable. Further, a cyclic error correcting code was designed and implemented to work with these cryptographic systems.

III. HISTORICAL BACKGROUND

Some of the earliest practical crytographic systems were the monoalphabetic substitution systems used by the Romans [Ref. 1]. In these, one letter is substituted for another. For example, an A might be replaced by a C. By the fifteenth century, an Italian by the name of Alberti came up with a technique of cryptoanalyzing letters by frequency analyses. As a result, he invented probably the first polyalphabetic substitution system using a cipher disk. Thus, he would rotate the disk and encode several more words with the next substitution alphabet.

Early in the sixteenth century Trithemius, a Benedictine Monk, had the first printed book published on cryptology. Trithemius described the square table or tableau which was the first known instance of a progressive key applied to polyalphabetic substitution. It provided a means of changing alphabets with each character. Later in the sixteenth century, Vigenere perfected the autokey; a progressive key in which the last decoded character led to the next substitution alphabet in a polyalphabetic key. These were basically the techniques that were widely applied in the cryptomachines in the first half of the twentieth century. Various transposition techniques have been employed including the wide use of changing word order and techniques such as rail transpositions (used in the Civil War).

In 1883, Auguste Kerckhoffs, a man born in Holland but a naturalized Frenchman, published a book entitled _La Cryptographic Militaire_. In it, he established two general principles for cryptographic systems. They were:

1. A key must withstand the operational strains of heavy traffic. It must be assumed that the enemy has the general system. Therefore, the security of the system must rest with the key.
2. Only cryptoanalysts can know the security of the key. In this, he infers that anyone who proposes a cryptographic technique should be familiar with the techniques that could be used to break it.

From these two general principles, six specific requirements emerged in his book:

1. The key should be, if not theoretically unbreakable, at least unbreakable in practice.
2. Compromise of the hardware system or coding technique should not result in compromising the security of communications that the system carries.
3. The key should be remembered without notes and should be easily changeable.
4. The cryptograms must be transmittable by telegraph. Today this would be expanded to include both digital intelligence and voice (if voice scramblers are employed) utilizing either wire or radio as the medium.

5. The apparatus or documents should be portable and operable by a single person.

6. The system should be easy, neither requiring knowledge of a long list of rules nor involving mental strain.

In 1917 Gilbert S. Vernam, a young engineer at American Telephone and Telegraph Company, using the Baudot code (teletype) invented a means of adding two characters (exclusive or). Vernam's machine mixed a key with text as illustrated by the following:

Clear Text	1	0	1	1	1
Key	0	1	0	1	0
Coded Character	1	1	1	0	1

To derive the text from the coded character, all that was required was the addition of the key again to the coded character.

Coded Character	1	1	1	0	1
Key	0	1	0	1	0
Clear Text	1	0	1	1	1

His machines used a key tape loop about eight feet long which caused the key to repeat itself over a high volume of traffic. This allowed cryptoanalysts to derive the key. William F. Friedman, in fact, solved cryptograms using single-loop code tapes but appears to have been

unsuccessful when two code tapes were used. Major Joseph Om Mauborgne (U.S. Army) then introduced the one-time code tape derived from a random noise source. This was one of the first theoretically (and in practice) unbreakable code systems. The major disadvantage of the system was the enormous amounts of key required for high-volume traffic.

During the 1920's and 1930's, the rotor-code machines having five and more rotors, each rotor representing a scrambling step, were developed. They proved relatively insecure, requiring only high-traffic volume for the cryptoanalyst to break them. In fact, the Japanese used a code-wheel-type machine for their diplomatic communications well into World War II. It was vulnerable to cryptoanalysis, and William F. Friedman and his group not only solved the code but reconstructed a model of the machine to break Japanese diplomatic correspondence. Thus, President Roosevelt and others were aware of the impending break in diplomatic relations with Japan just prior to World War II.

The code wheels (or rotors) were nothing more than key memories storing quantities of key which could easily be changed by interchanging rotor positions, specifying various start points for each rotor, and periodicaly replacing a set of rotors. This provided a means of producing what is called key leverage.

The advent of electronic enciphering systems substantially replaced the mechanical cryptographic machines. And, further the appearance and fast development of digital logic is offering new tools to modern crypto designers. References (2), (3) and (4) from the Bell System Technical Journal provide interesting literature on Digital Data Scramblers.

Today, the most commonly encountered commercial cryptosystem is based on the "shift register," [Ref. 5]. Despite design variations, shift registers are used as pseudorandom key generators. The implementation of data scramblers with pseudorandom sequences using logic circuits is suggested by Twigg [Ref. 6], and Henrickson [Ref. 7]. The idea of shift register sequences is well treated by Golomb [Ref. 8]. The relative weakness of pseudorandom codes is pointed by Meyer and Tuchman [Ref. 9], from I.B.M. For high security, Torrieri [Ref. 10], and Geffe [Ref. 11], introduce the idea of using nonlinear as well as linear operations. The theory of nonlinear operations is also contained in Ref. 8.

Finally, the appearance of modern high speed digital computers has risen speculation as how best to apply its capabilities since it is available for both cryptography and cryptanalysis. Even the newest microprocessors are reported [Ref. 12], as being designed for encription devices.

A very comprehensive historical exposition with some descriptive technical content is the book by Kahn, <u>The</u>

Codebreakers [Ref. 13], which appeared in 1967. Of special interests are the sections devoted to the cryptographic agencies of the major powers, including the United States.

For the interested reader in the field of cryptography, the American Cryptogram Association publishes "The Cryptogram," a bimonthly magazine of articles and cryptograms. The hobby of solving cryptograms provides a fascinating intellectual challenge. Patient analysis and flashes of insight, combined with the enthusiasm of uncovering something hidden, give cryptanalysts an enjoyment which is almost unique.

IV. THEORY OF SECRECY SYSTEMS

A. INTRODUCTION

A secrecy system is defined as a set of transformations of one space (the set of possible messages) into a second space (the set of possible cryptograms). Each particular transformation of the set corresponds to enciphering with a particular key. The transformations are supposed reversible (non-singular) in order to obtain unique deciphering when the key is known together with the specific system used.

Each key and therefore each transformation is assumed to have an a priori probability associated with it. Similarly each possible message is assumed to have an associated a priori probability of being selected for encryption. These two represent the a priori knowledge of the situation for a cryptoanalyst trying to break the cipher.

To use the system a key is first selected and sent to the receiving point. The choice of a key determines a particular transformation in the set forming the system. Then a message is selected and the particular transformation corresponding to the selected key is applied to the message to produce a cryptogram. This cryptogram is transmitted to the receiving point by a channel where it can be intercepted by an undesired agent. At the receiving end, the inverse of the particular transformation is applied to the cryptogram

to recover the original message. Figure 1 provides the conceptual idea of a secrecy system.

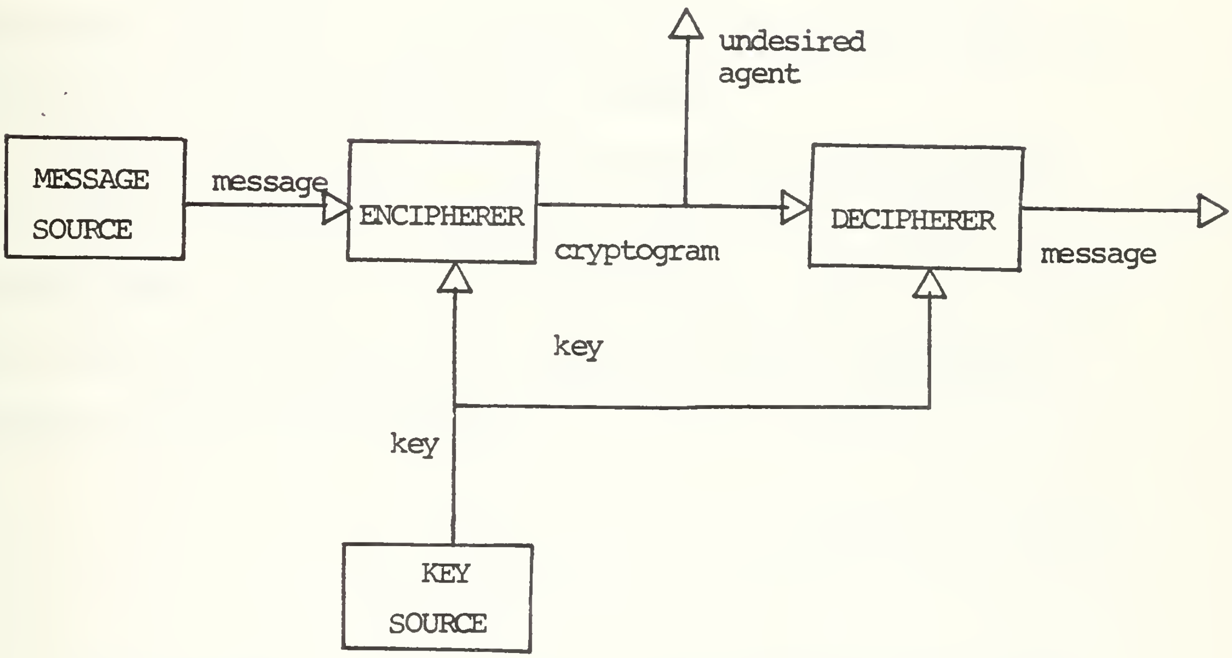


Figure 1. A Secrecy System.

If the referred undesired agent intercepts the transmitted cryptogram through a channel, he can calculate from it and from his possibel knowledge of the system being used, the a posteriori probabilities of the various possible messages and keys which might have produced this cryptogram. This set of a posteriori probabilities constitutes his knowledge of the key and message after the interception.

The calculation of the a posteriori probabilities is the generalized problem in cryptanalysis.

C. PERFECT SECRECY

Shannon [Ref. 14], provides for concepts such as entropy, redundancy, equivocation and many others that are helpful for evaluating secrecy systems.

Let us assume that the message space is constituted by a finite number of messages P_1, P_2, ..., P_n with an associated a priori probabilities $p(P_1)$, $p(P_2)$, ..., $p(P_n)$ and that these messages are mapped into the cryptogram space by the transformation

$$C_j = T_i \; P_j$$

The cryptanalyst intercepts a particular C_j and can then calculate the a posteriori conditional probability for the various messages, $p(P_j/C_j)$. It seems natural now to define that one condition for perfect secrecy is that for all C_j, the a posteriori probabilities of the messages P given that C_j has been received, are equal to their a priori probabilities, independent of these values. Or, from an information theory viewpoint, intercepting the cryptogram has given the cryptanalyst no information about the message; he just knows that a message was sent. On the other hand, if this condition is not satisfied there will exist situations in which the cryptanalyst has certain

a priori probabilities and certain choices of key and message thus preventing perfect secrecy to be achieved.

Shannon [Ref. 15], gives a theorem stating the necessary and sufficient conditions for perfect secrecy, namely

$$p(C/P) = p(C)$$

for all the messages (P) and all the cryptograms (C). Where

$p(C/P)$ = Conditional probability of cryptogram C to occur if message P is chosen.

$p(C)$ = Probability of obtaining cryptogram C for any cause.

Stated in other terms, the total probability of all keys that transform P_i into a given cryptogram C is equal to that of all keys transforming P_j into the same C, for all P_i, P_j and C.

In the Mathematical Theory of Communications given by Reference 14, it was shown that a convenient measure of information was the entropy. For a set of events with probabilities p_1, p_2, ..., p_n, the entropy H is given by:

$$H = - \sum_{n} p_i \log p_i$$

In a secrecy system there are two choices involved, that of the message and that of the key. We may measure the amount of information produced when a message is chosen by

$$H(P) = - \Sigma p(P) \log p(P)$$

the summation being over all possible messages. Similarly, there is an uncertainty associated with the choice of key given by

$$H(K) = - \Sigma p(K) \log p(K)$$

For perfect secrecy systems the amount of information in the message is at most log n (occurring when all messages are equiprobable). This information can be concealed completely only if the key uncertainty is at least log n. In a more general way of expressing this: There is a limit to what we can achieve with a given uncertainty in key, the amount of uncertainty we can introduce into the solution cannot be greater than the key uncertainty.

The situation gets more complicated if the number of messages is infinite. For example, assume that messages are generated as infinite sequences of letters by a suitable Markoff process. From the definition, no finite key will give perfect secrecy. We can suppose then, that the key source generates keys in the same manner, that is as an

infinite sequence of symbols. Suppose further that only a certain length L_k is needed to encipher and decipher a length L_p of message. Let the logarithm of the number of letters in the message alphabet be R_p and that for the key alphabet be R_k. Then from the finite case, it is evident that perfect secrecy requires

$$R_p L_p \leq R_k L_k$$

This type of perfect secrecy is obtained by the Vernam system [Ref. 16].

Thus, it can be concluded that the key required for perfect secrecy depends on the total number of possible messages. The disadvantage of perfect systems for large correspondence systems such as for data communications and data retrieval services, is the equivalent amount of key that must be sent.

In this paper the requirement for a large key for large messages is eliminated by designing a self keyed system that will continually originate key letters based on several past letters that were already ciphered. Provided enough distance is chosen in between selected letters the system will avoid the statistical dependency of consecutive letters in a natural language, thus generating a sequence of key letters suitable for any message length.

D. EQUIVOCATION

A cryptographic system can be compared with a communication system in the sense that whereas in one the signal is unintentionally perturbed by noise, and in the other, namely the cryptographic system, the message is intentionally perturbed by the ciphering process to hide the information. Thus, there is an uncertainty of what was actually transmitted. From information theory a natural mathematical measure of uncertainty is the conditional entropy of the transmitted signal when the received signal is known. This conditional entropy is known as equivocation.

$$H(X/Y) = -\Sigma\, p(x,y) \log p(x/y)$$

From the point of view of the cryptanalyst, a secrecy system is almost identical with a noisy communication system. The message is operated by a statistical element, the enciphering system, with its statistically chosen key. The result of this operation is the cryptogram, which when transmitted is vulnerable to interception and available for analysis. The main differences in the two cases are:

1. The operation of the enciphering transformation is generally of a more complex nature than the perturbing noise in a channel.

2. The key for a secrecy system is usually chosen from a finite set of possibilities while the noise in the

channel is more often continually introduced, in effect chosen from an infinite set.

With these considerations in mind it is natural to use the equivocation as a theoretical secrecy index. It may be noted that there are two significant equivocations, that of the key and that of the message which are denoted as H(K/C) and H(P/C):

$$H(K/C) = -\Sigma\, p(C,K) \log p(K/C)$$

$$H(P/C) = -\Sigma\, p(C,P) \log p(K/P)$$

The same general arguments used to justify the equivocation as a measure of uncertainty in communication theory apply here as well. Zero equivocation requires that one message (or key) have unit probability and all others zero, corresponding to complete knowledge.

E. IDEAL SECRECY SYSTEMS

In Reference 15, the concept of equivocation leads to means of evaluating secrecy systems as a function of the amount of N, the number of letters received. It is shown that for most systems as N increases the referred equivocations tend to decrease to zero, consequently the solution of the cryptogram becomes unique at a point called unicity point.

In the section on Perfect Secrecy it was stated that perfect secrecy requires an infinite amount of key if

messages of unlimited length are allowed. With a finite key size, the equivocation of key and message generally approaches zero. The other extreme is for H(K/C) to be equal to H(K). Then, no matter how much material is intercepted, there is not a unique solution but many of comparable probability. An ideal system can be defined as one in which H(K/C) and H(P/C) do not approach zero as N increases. A strongly ideal system would be one in which H(K/C) remains constant at H(K), that is, knowing the cryptogram has not aided in solving the key uncertainty.

An example of an ideal cipher is a simple substitution in an artificial language in which all letters are equiprobable and successive letters independently chosen.

With natural languages it is in general possible to approximate the ideal characteristic. The complexity of the system needed usually goes up rapidly when an attempt is made to realize this. To approximate the ideal equivocation, one may first operate on the message with a transducer which removes all redundancies. After this almost any simple ciphering system – substitution, transposition, etc., is satisfactory. The more elaborate the transducer and the nearer the output is to the desired form, the more closely will the secrecy system approximate the ideal characteristic.

The work to be presented in following sections, will describe a scheme to approximate the ideal secrecy system by using a digital computer to mainly accomplish two things:

1. Change the probability structure of natural languages to obtain an almost equiprobable occurrence of letters.

2. Eliminate the statistical dependence of successive letters in natural languages.

Further, a message transformed to reflect these properties, will be either transmitted as such or an additional conventional ciphering can be made.

V. DIGITAL SUBSTITUTION

The development of a digital substitution cipher was the first step taken to accomplish the present work. After it, more complex variations were experimented to obtain a reasonable secure system taking advantage of the use of the computer. Thus, it can be said that most of the subsequent work rests on these first results. A brief explanation follows of the Decwriter system and its character codes used to interface with the PDP-11/40 computer.

A. THE DECWRITER SYSTEM

The LC11 Decwriter system is a high-speed teletype-writer designed to interface with the PDP-11 family of processors to provide both: Input (keyboard) and output (printer) functions for the system. It can be used as the console input/output device. The system can receive characters from the keyboard or can print at speeds up to 30 characters per second in standard ASCII formats. The character code used is USASCII-68 which is listed in Table No. I. From these 128 characters, only 64 are printing characters, those of columns 2, 3, 4 and 5. Table No. II presents these 64 characters and their correspondent binary representation.

ROW	COLUMN BITS 4321 / 765	0 000	1 001	2 010	3 011	4 100	5 101	6 110	7 111
0	0000	NUL	DLE	SP	0	@	P	`	p
1	0001	SOH	DC1	!	1	A	Q	a	q
2	0010	STX	DC2	"	2	B	R	b	r
3	0011	ETX	DC3	#	3	C	S	c	s
4	0100	EOT	DC4	$	4	D	T	d	t
5	0101	ENQ	NAK	%	5	E	U	e	u
6	0110	ACK	SYN	&	6	F	V	f	v
7	0111	BEL	ETB	'	7	G	W	g	w
8	1000	BS	CAN	(	8	H	X	h	x
9	1001	HT	EM	)	9	I	Y	i	y
10	1010	LF	SUB	*	:	J	Z	j	z
11	1011	VT	ESC	+	;	K	[	k	{
12	1100	FF	FS	,	<	L	\	l	¦
13	1101	CR	GS	–	=	M	]	m	}
14	1110	SO	RS	.	>	N	⌒	n	~
15	1111	SI	US	/	?	O	—	o	DEL

TABLE I - USASCII-68 CHARACTER CODE

<table>
<tr><td>SP</td><td>10100000</td><td>0</td><td>10110000</td><td>@</td><td>11000000</td><td>P</td><td>11010000</td></tr>
<tr><td>!</td><td>10100001</td><td>1</td><td>10110001</td><td>A</td><td>11000001</td><td>Q</td><td>11010001</td></tr>
<tr><td>"</td><td>10100010</td><td>2</td><td>10110010</td><td>B</td><td>11000010</td><td>R</td><td>11010010</td></tr>
<tr><td>#</td><td>10100011</td><td>3</td><td>10110011</td><td>C</td><td>11000011</td><td>S</td><td>11010011</td></tr>
<tr><td>$</td><td>10100100</td><td>4</td><td>10110100</td><td>D</td><td>11000100</td><td>T</td><td>11010100</td></tr>
<tr><td>%</td><td>10100101</td><td>5</td><td>10110101</td><td>E</td><td>11000101</td><td>U</td><td>11010101</td></tr>
<tr><td>&</td><td>10100110</td><td>6</td><td>10110110</td><td>F</td><td>11000110</td><td>V</td><td>11010110</td></tr>
<tr><td>'</td><td>10100111</td><td>7</td><td>10110111</td><td>G</td><td>11000111</td><td>W</td><td>11010111</td></tr>
<tr><td>(</td><td>10101000</td><td>8</td><td>10111000</td><td>H</td><td>11001000</td><td>X</td><td>11010000</td></tr>
<tr><td>)</td><td>10101001</td><td>9</td><td>10111001</td><td>I</td><td>11001001</td><td>Y</td><td>11011001</td></tr>
<tr><td>*</td><td>10101010</td><td>:</td><td>10111010</td><td>J</td><td>11001010</td><td>Z</td><td>11011010</td></tr>
<tr><td>+</td><td>10101011</td><td>;</td><td>10111011</td><td>K</td><td>11001011</td><td>[</td><td>11011011</td></tr>
<tr><td>,</td><td>10101100</td><td><</td><td>10111100</td><td>L</td><td>11001100</td><td>|</td><td>11011100</td></tr>
<tr><td>-</td><td>10101101</td><td>=</td><td>10111101</td><td>M</td><td>11001101</td><td>]</td><td>11011101</td></tr>
<tr><td>.</td><td>10101110</td><td>></td><td>10111110</td><td>N</td><td>11001110</td><td>^</td><td>11011110</td></tr>
<tr><td>/</td><td>10101111</td><td>?</td><td>10111111</td><td>O</td><td>11001111</td><td>_</td><td>11011111</td></tr>
</table>

TABLE II - DECWRITER PRINTING CHARACTERS AND THEIR BINARY REPRESENTATION

B. APPLICATION OF GROUP THEORY TO CRYPTOGRAPHY

A group is defined as a set of elements a, b, c, ... and an operation, denoted by + for which the following properties are satisfied:

a) For any elements a,b, in the set, a + b is in the set.

b) The associative law is satisfied; that is, for any a,b,c in the set

$$a + (b + c) = (a + b) + c$$

c) There is an identity element, I, in the set such that

$$a + I = I + a = a; \quad \text{all } a \text{ in the set.}$$

d) For each element a, there is an inverse a^{-1} in the set satisfying

$$a + a^{-1} = a^{-1} + a = I$$

A group is abelian or commutative if

$$a + b = b + a \quad \text{for all } a \text{ and } b \text{ in the set.}$$

The integers under ordinary addition and the set of binary sequences of a fixed length n under exclusive-or operation are examples of abelian groups.

From boolean algebra, an additional property of an abelian group of binary sequences of a fixed length n under the exclusive-or operation is that,

given	$a + b = c$	
then	$a + c = b$	
and	$b + c = a$;	for all a,b and c in the group.

The 8-bit binary sequences with which the computer handles the ASCII code characters is in this sense an abelian group. This last property suggested the idea of encrypting simply by exclusive-oring the desired set of sequences by a key (another sequence or a set of sequences). Decrypting or recovery of the original sequences can be done simply by exclusive-oring the obtained set of sequences with the key.

Basically the transformation can be expressed as

$C = K + P$, for encryption, and

$P = K + C$, for decryption,

where C, K and P represent an 8-bit sequence stored in a register and the symbol + stands for the logical exclusive-or operation.

While it is clear that the whole 2^8 8-bit sequences can be used to represent crypto sequences, since this set

of sequences constitute an abelian group; a limitation was imposed through this work to allow transformations to be done between printing characters (those of Table II). That is, restrict the domain and range of the transformations to the binary sequences of Table II.

We can further realize the 12 possible combinations of two sequences of same or different sets by exclusive-oring them and observe that the range of the transformations is given by the sets of sequences whose 4-left most are:

0 0 0 0	for	A + A
		B + B
		C + C
		D + D
0 0 0 1	for	A + B
		B + A
		C + D
		D + C
0 1 1 0	for	A + C
		C + A
		B + D
		D + B
0 1 1 1	for	A + D
		D + A
		B + C
		C + B

C. TRANSFORMATIONS

From Table II it can be observed that these sequences no longer form a group under the exclusive-or operation, since choosing any two sequences will originate a new sequence not in the referred table. For example:

$$
\begin{array}{rl}
\text{Plaintext character} = A = & 1\,1\,0\,0\,0\,0\,0\,1\ + \\
\text{Key character} = L = & \underline{1\,1\,0\,0\,1\,1\,0\,0} \\
\text{Ciphered character} = & 0\,0\,0\,0\,1\,1\,0\,1
\end{array}
$$

And we obtained a sequence 0 0 0 0 1 1 0 1 not in the table.

If we observe sets A, B, C and D of Table II, we will observe that each set has its 4-left most bits equal. Or that the dom ain of the transformation is given by the sequences whose 4-left most bits are:

Set A	1 0 1 0
Set B	1 0 1 1
Set C	1 1 0 0
Set D	1 1 0 1

In order to make the range of the transformations equal to its domain in accordance with the restriction imposed, an additional binary multiplier: The intermediate key (IK) was devised. It allowed for mapping into the 64 printing characters.

The value of IK is dependent on the particular transformation desired and the key to be used. For example: A system is designed to transform characters from set B into characters of set C for encryption. The decryption is done by doing the inverse. Now assume that the key to be used for a particular transformation belongs to set D.

Plaintext character	= 8 =	10111000	(Set B)
Key character	= Z =	<u>11011010</u>	(Set D)
		01100010	
	IK =	<u>10100000</u>	
Crypto character	= B =	11000010	(Set C)

The intermediate key value was obtained by exclusive-oring the 4-left most bits of the plaintext, the key and the crypto characters, as shown below.

Plaintext character	1011 +
Key character	1101 +
Crypto character	<u>1100</u>
IK	10100000

For decrypting the inverse is done, that is:

Crypto character = B = 11000010 (Set C)

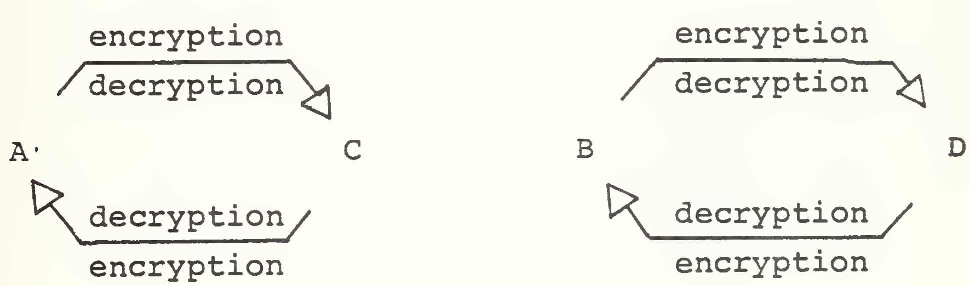
Key character = Z = 11011010 (Set D)

00011000

IK = 10100000

Plaintext character = 8 = 10111000

Based on the concepts so far presented and the idea of the intermediate key multiplier, that allows for sequences of Table II to behave like a group, Table III was constructed. It gives the necessary values of IK for all possible transformations in between sets. From this general table, it can be obtained typical tables of required values of IK for each specific transformation. For example, if we assume that the desired transformation between the four sets were



Then the required table of IK values will be:

		KEY SET															
		A				B				C				D			
		CRYPTOCHARACTER SET															
		A	B	C	D	A	B	C	D	A	B	C	D	A	B	C	D
Plaintext character set	A	A	B	C	D	B	A	D	C	C	D	A	B	C	D	A	B
	B	B	A	D	C	A	B	C	D	D	C	B	A	C	D	A	B
	C	C	D	A	B	D	C	B	A	A	B	C	D	B	A	D	C
	D	D	C	B	A	C	D	A	B	B	A	D	C	A	B	C	D

Where: A = 10100000
B = 10110000
C = 11000000
D = 11010000

* This is not the ASCII representation of A, B, C, D but the binary string that they represent in the table.

TABLE III - INTERMEDIATE KEY VALUES

		KEY SET			
		A	B	C	D
PLAINTEXT SET	A	C	D	A	B
	B	C	D	A	B
	C	C	D	A	B
	D	C	D	A	B

D. SIMPLE SUBSTITUTION

Although the scheme developed and presented until now provides for transformations using the 64 printing characters, a restriction was placed to be able to handle only the 26 letters of the English alphabet plus the additional 6 characters that appear in Table No. II, sets C and D. Thus, for the simple substitution ciphers transformations were designed between these two sets, that is,

$$\begin{tikzcd} C \arrow[r, bend left, "\text{Encryption}", "\text{Decryption}"'] & D \arrow[l, bend left, "\text{Decryption}", "\text{Encryption}"'] \end{tikzcd}$$

And the corresponding table of values of intermediate keys will be:

PLAINTEXT SET	KEY SET A	KEY SET B	KEY SET C	KEY SET D
A	B	A	D	C
B	B	A	D	C
C	B	A	D	C
D	B	A	D	C

Figure 2 shows in block diagram the computer realization of this simple substitution cipher. Appendix A gives the complete program to accomplish this. Figure 3 is an example of this cipher.

E. GRAPHICAL REPRESENTATION OF RESULTS

Natural languages, such as English, Spanish, German, French, etc., have a characteristic letter frequency. For example, the normal frequency for English is as shown in Table IV.

For the purpose of observing the statistical nature of plaintexts as well as of cryptograms obtained, a computer program (shown in Appendix B and C) was made to realize the following computations:

- Count the number of occurrences of each letter in a text.
- Calculate and plot the percentage of occurrence of each character in the text.
- Calculate the mean value of percentage of occurrences.
- Calculate the standard deviation of the percentage of occurrences.

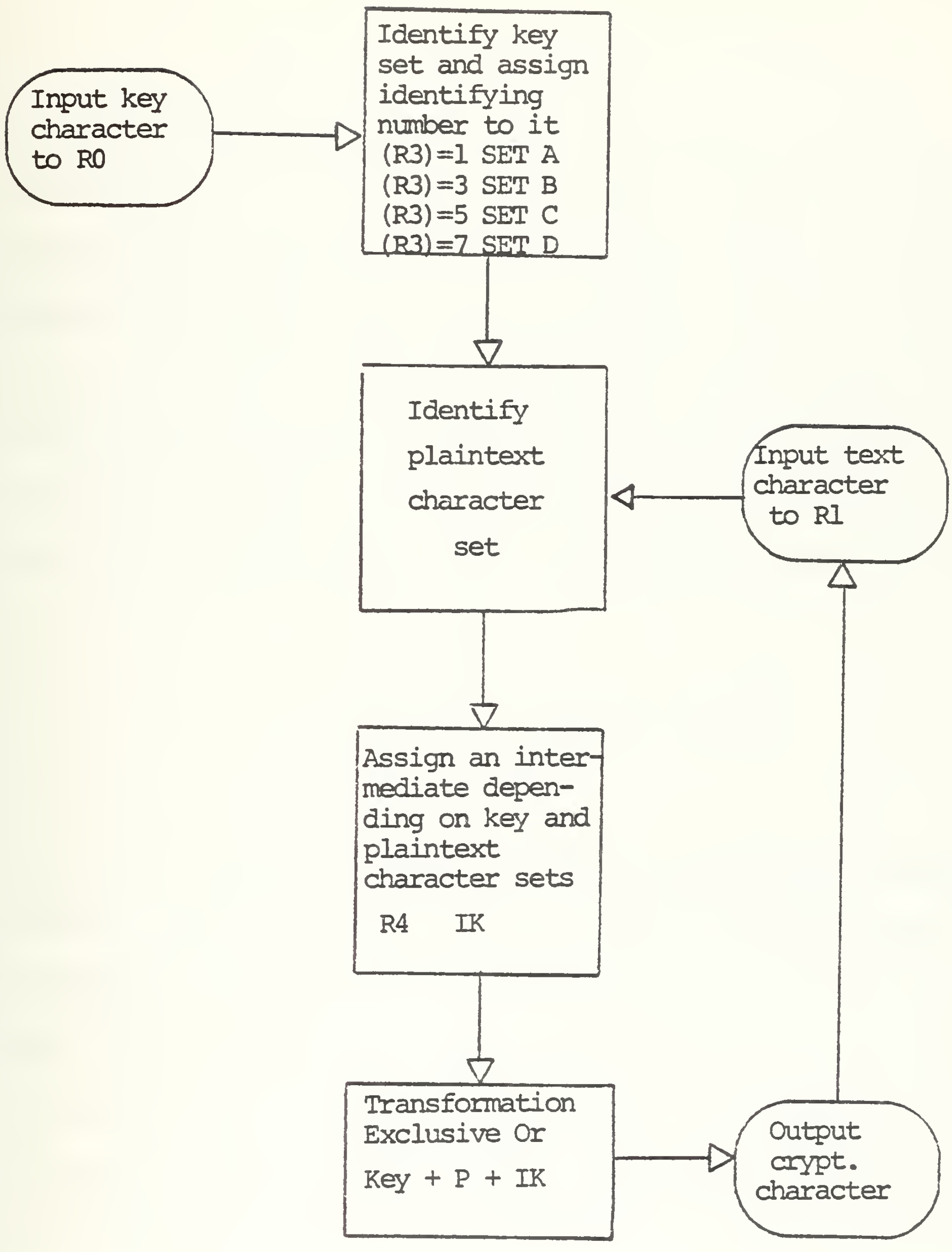


Figure 2. Block diagram of the program for the simple substitution cipher

```
NTHIS_BOOK_IS_DESIGNED_PRIMARILY_FOR_USE_AS_A_FIRST_YEAR
..GRADUATE_TEXT_IN_INFORMATION_THEORY_SUITABLE_FOR_BOTH_E
NGINEERS_AND_MATHEMATICIANS_@_IT_IS_ASSUMED_THAT_THE_RER
DER_HAS_SOME_UNDERSTANDING_OF_FRESHMAN_CALCULUS_AND_ELEM
ENTARY_PROBABILITY_AND_IN_THE_LATER_CHAPTERS_SOME_INTROD
UCTORY_RANDOM_PROCESS_THEORY_@_UNFORTUNATELY_THERE_IS_ON
E_MORE_REQUIREMENT_THAT_IS_HARDER_TO_MEET_@_THE_READER_M
UST_HAVE_A_REASONABLE_LEVEL_OF_MATHEMATICAL_MATURITY
```

a) Plaintext message (input)

```
NC_^DHUXX\H^DHSRD^PYRSHGE^ZVE^[NHQXEHBDRHVDHVHQ^EDCHNRVE
HPEVSBVCRHCROCH^YH^YQXEZVC^XYHC_PXENHDB^CVU[RHQXEHUXC_HR
YP^YRREDHVYSHZVC_RZVC^T^VYDHWH^CH^DHVDDBZRSHC_VCHC_RHERV
SREH_VDHDXZRHBYSREDCVYS^YPHXQHQERD_ZVYHTV[TB[EDHVYSHR[RZ
RYCVENHGEXUVU^[^CNHVYSH^YHC_RH[VCREHT_VGCREDHDXZRH^YCEXS
BTCXENHEVYSXZHGEXTRDDHC_RXENHWHBYQXECBYVCR[NHC_RERH^DHXY
RHZXERHERFB^ERZRYCHC_VCH^DH_VESREHCXHZRRCHWHC_RHERVSREHZ
BDCH_VARHVHERVDXYVU[RH[RAR[HXQHZVC_RZVC^TV[HZVCBE^CN
```

b) Cryptogram message (output)

Figure 3. Example of a simple substitution cipher: Encrypting process. Key = W

Alphabetically		By frequency	
A	7.3%	E	13.0%
B	0.9	T	9.3
C	3.0	N	7.8
D	4.4	R	7.7
E	13.0	I	7.4
F	2.8	O	7.4
G	1.6	A	7.3
H	3.5	S	6.3
I	7.4	D	4.4
J	0.2	H	3.5
K	0.3	L	3.5
L	3.5	C	3.0
M	2.5	F	2.8
N	7.8	P	2.7
O	7.4	U	2.7
P	2.7	M	2.5
Q	0.3	Y	1.9
R	7.7	G	1.6
S	6.3	W	1.6
T	9.3	V	1.3
U	2.7	B	0.9
V	1.3	X	0.5
W	1.6	K	0.3
X	0.5	Q	0.3
Y	1.9	J	0.2
Z	0.1	Z	0.1

TABLE IV - FREQUENCY OF THE LETTERS OF THE ENGLISH ALPHABET, ARRANGED ALPHABETICALLY AND BY FREQUENCY

For each transformation done, the text was analyzed by this program and the results were plotted. In the horizontal axis are the 32 chosen characters in the following order from zero to 31:

@ A B C D E F G H I J K L M N O P Q R S T U V W X Y Z [/] ^ _

In the vertical axis the percentage of occurrence scale or frequency distribution is plotted.

Examples of these plots are given by Figures 5 to 8. There the frequency distribution of letters for the following languages is plotted:

Figure 4: ENGLISH

Figure 5: SPANISH

Figure 6: FRENCH

Figure 7: ITALIAN

The author has preferred to give the results achieved through this work by presenting these plots rather than giving messages and their cryptograms as examples of what was obtained. Inherent with these plots is an evaluation of the system used in each case. Additional information that will be found in these plots is the standard deviation of percentage of occurrence of the character in each cryptogram.

For the simple substitution cipher, it was expected to obtain similar results as for the plaintext of Figure 5. Figures 8 to 10 show the frequency distribution of characters when this system was used with different keys. As expected, similar results were obtained but with the values changed from one character to another. This occurred since one character or letter has just been replaced by another through these transformations. Table V presenting in tabular form the number of occurrences for these substitutions gives a figure of what has occurred with the messages in each case.

In Section IV, Theory of Secrecy Systems, it was stated that one goal to achieve ideal secrecy was to change the probability structure of natural languages to obtain an equiprobable occurrence of letters. This is the reason why the calculation of standard deviation was considered to evaluate secrecy obtained. Since the language to be used in this present work will be English it may be useful to keep in mind that the standard deviation for an English text is 3.81 as stated in Figure 4.

F. PSEUDORANDOM SUBSTITUTION

The simple substitution cipher can also be called monoalphabetic cipher since there is only one alphabet to encipher the message. The cryptanalytic weakness of this cipher is the fact that a given plain language letter is always represented by the same crypto letter.

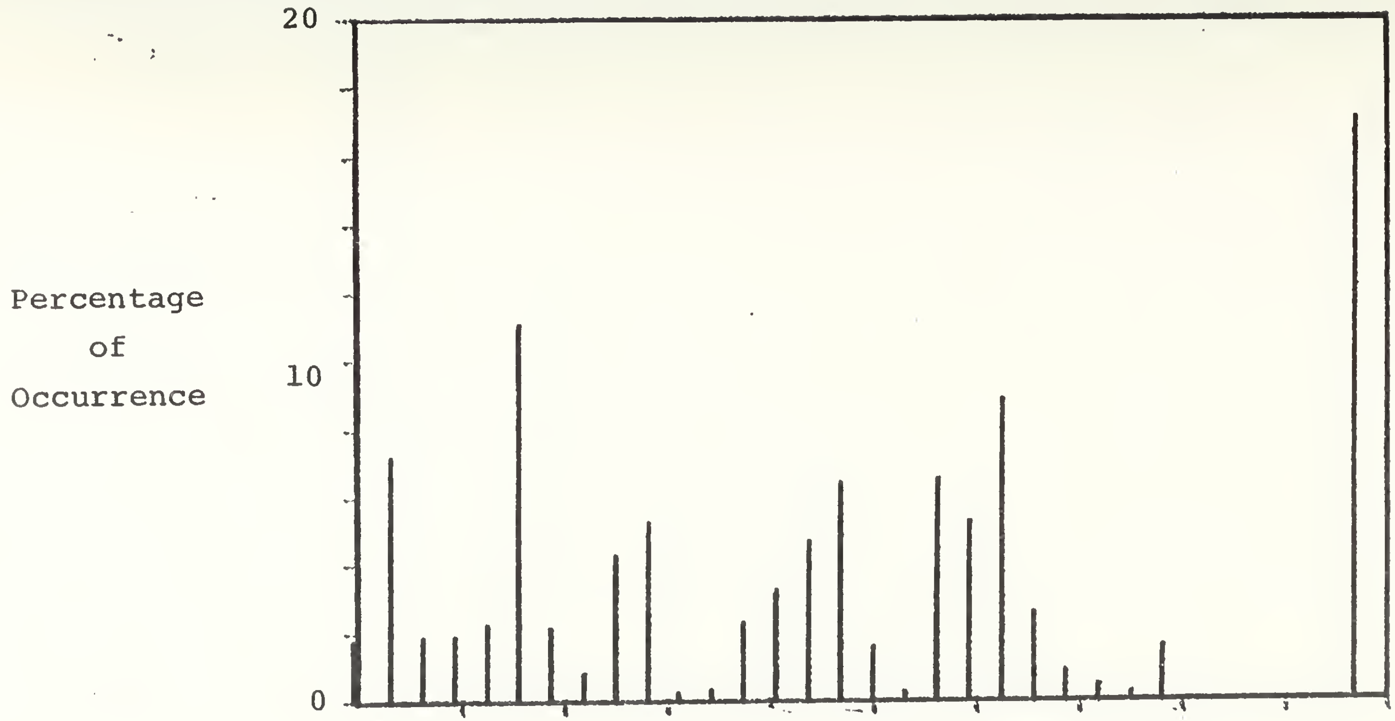


Figure 4. Plaintext English Language

Standard deviation = 3.81

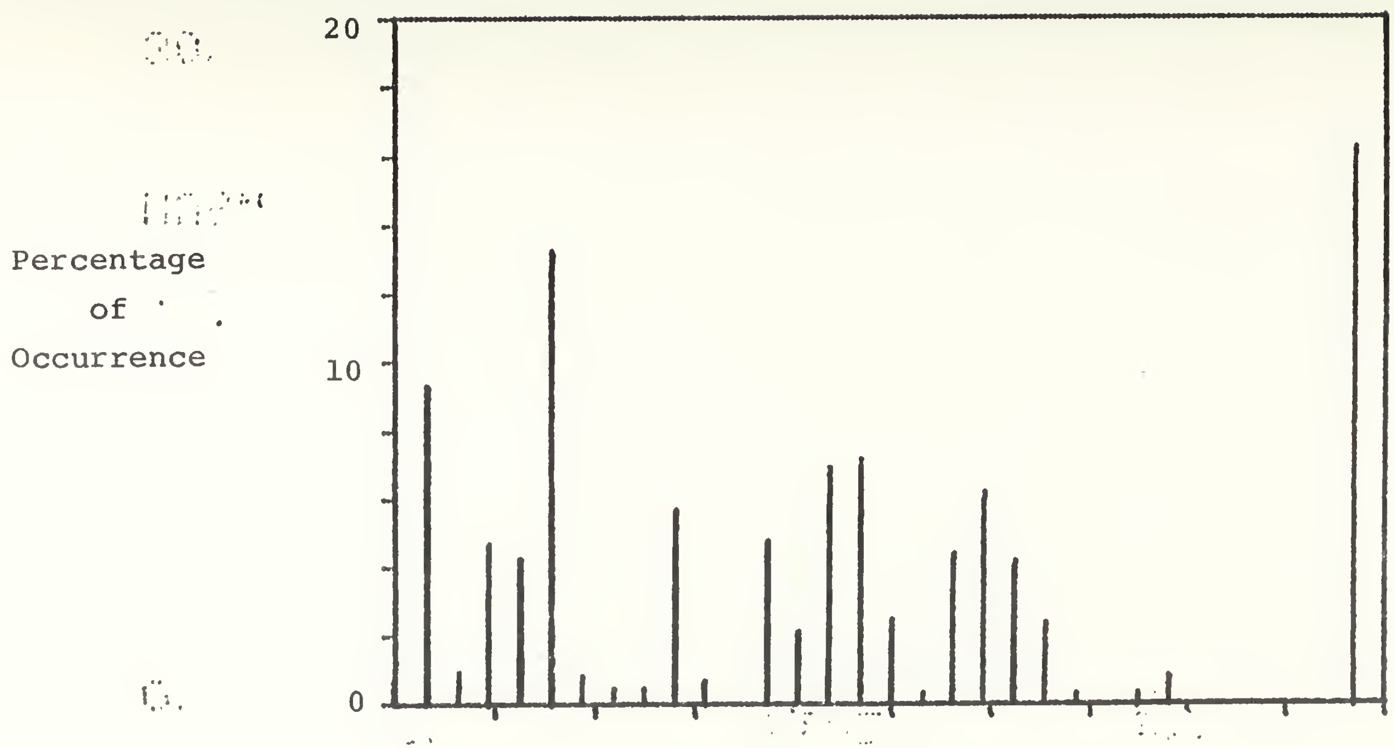


Figure 5. Plaintext Spanish Language
Standard deviation = 3.972

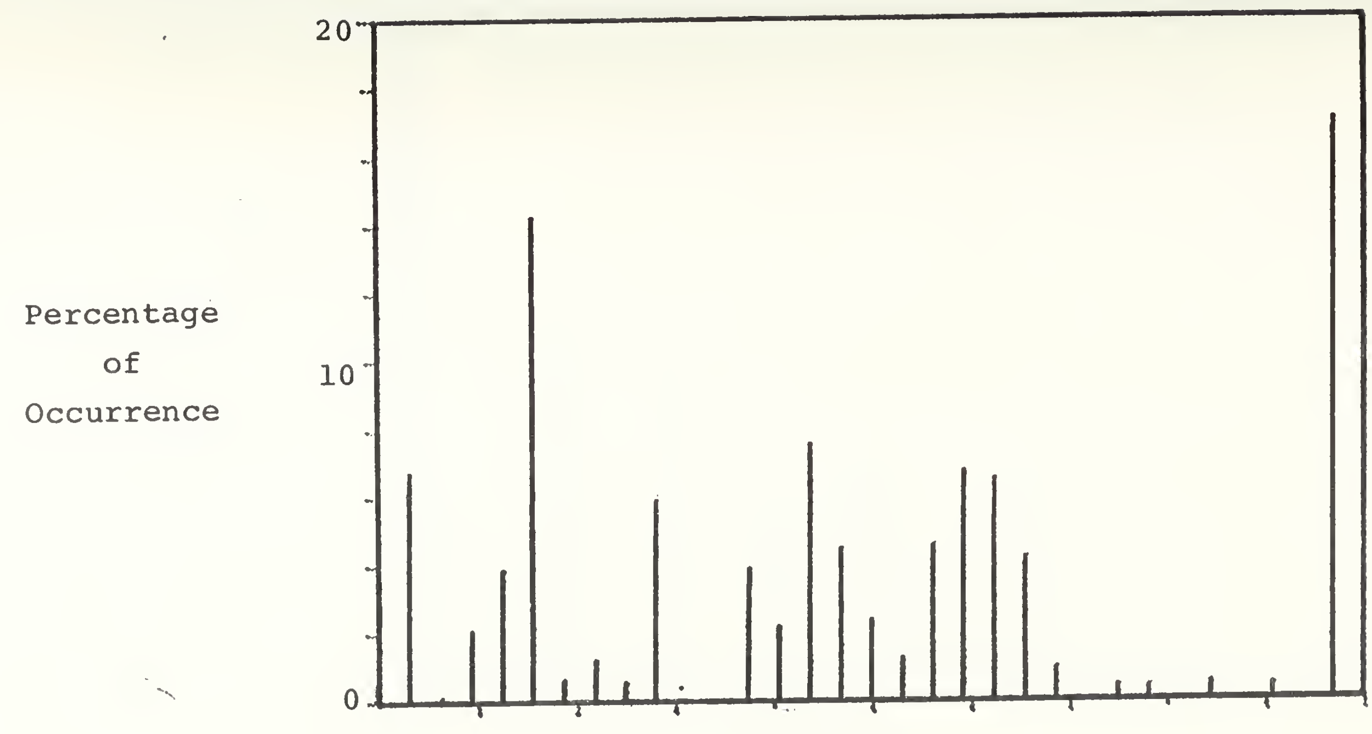


Figure 6. Plaintext French Language
Standard deviation = 4.037

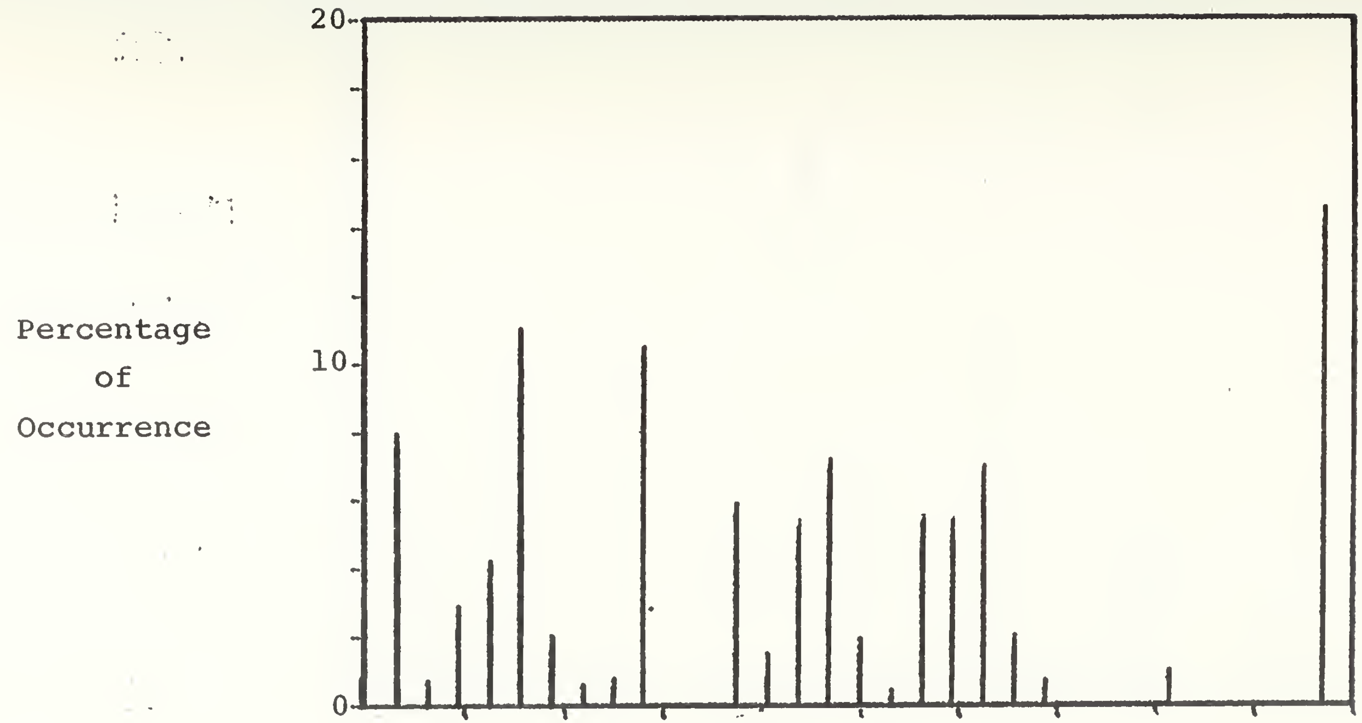


Figure 7. Plaintext Italian Language
Standard deviation = 3.873

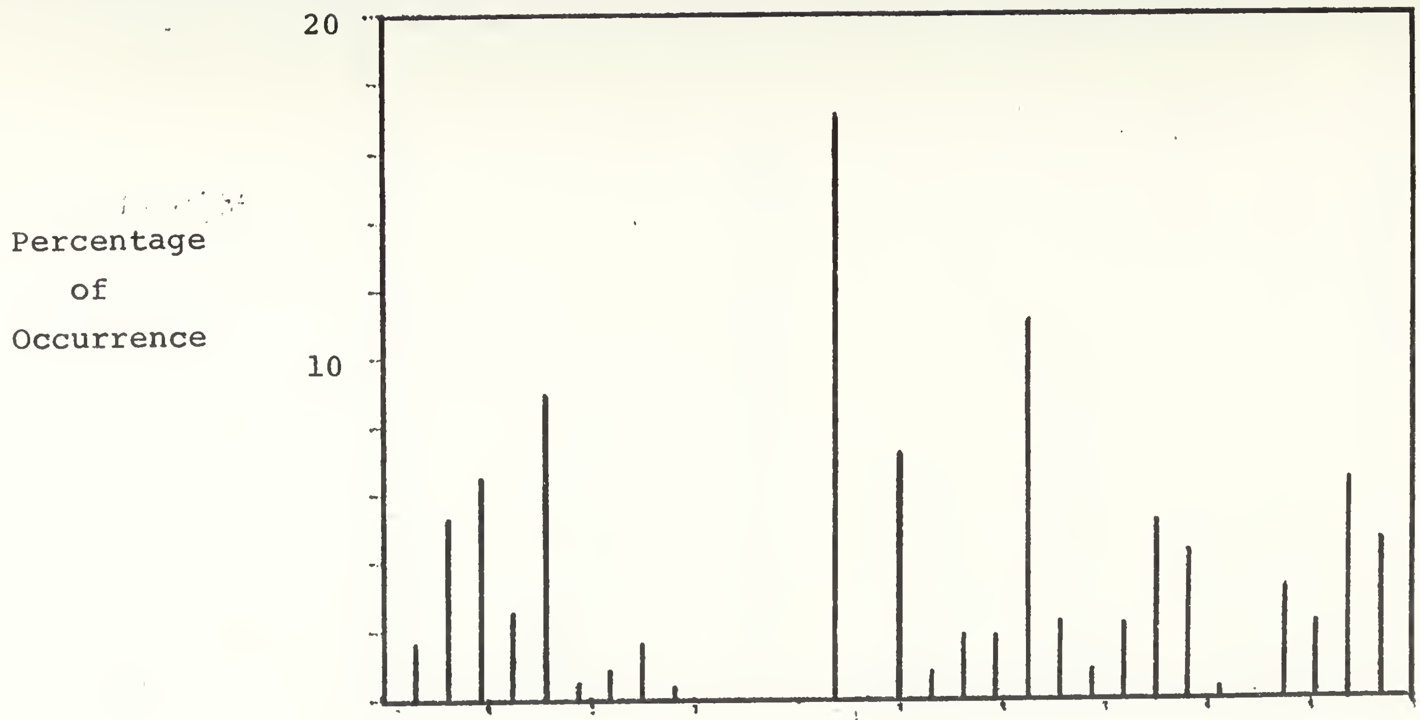


Figure 8. Simple substitution cipher
Standard deviation = 3.81
Key = A

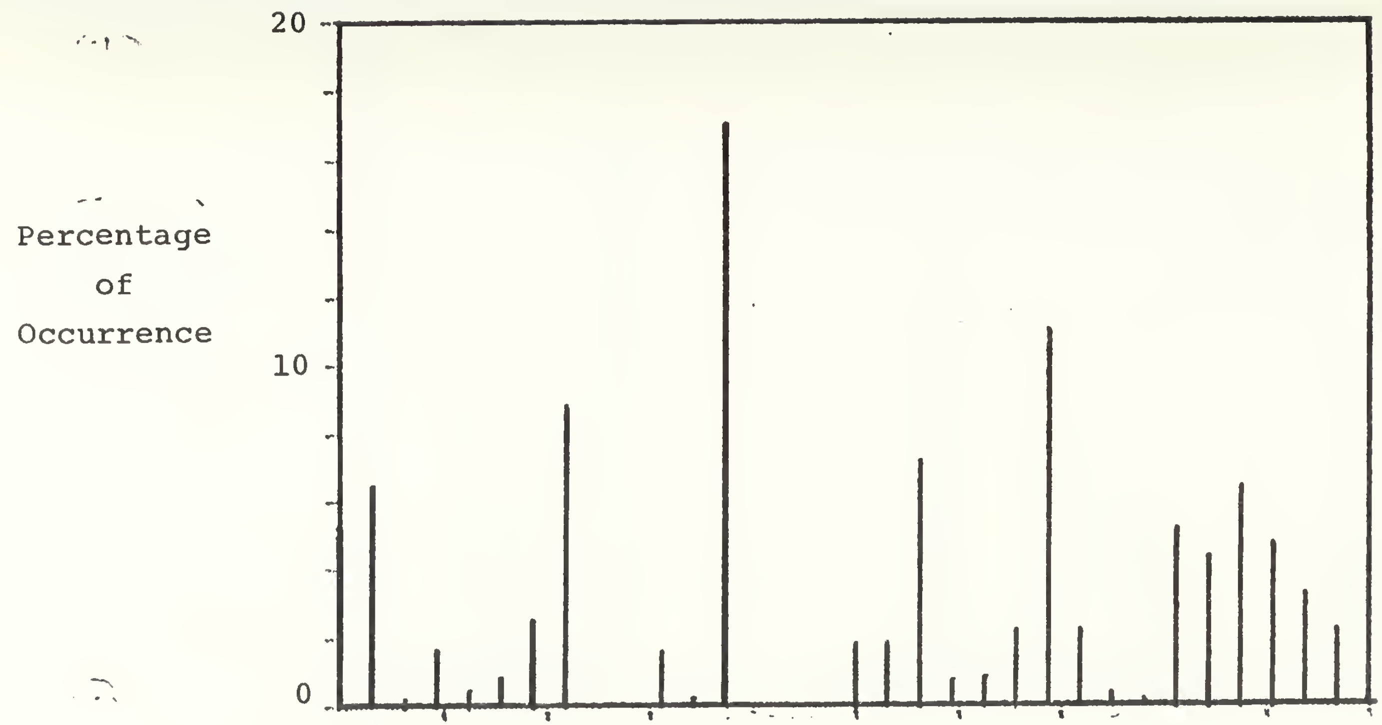


Figure 9. Simple substitution cipher
Standard deviation = 3.81
Key = C

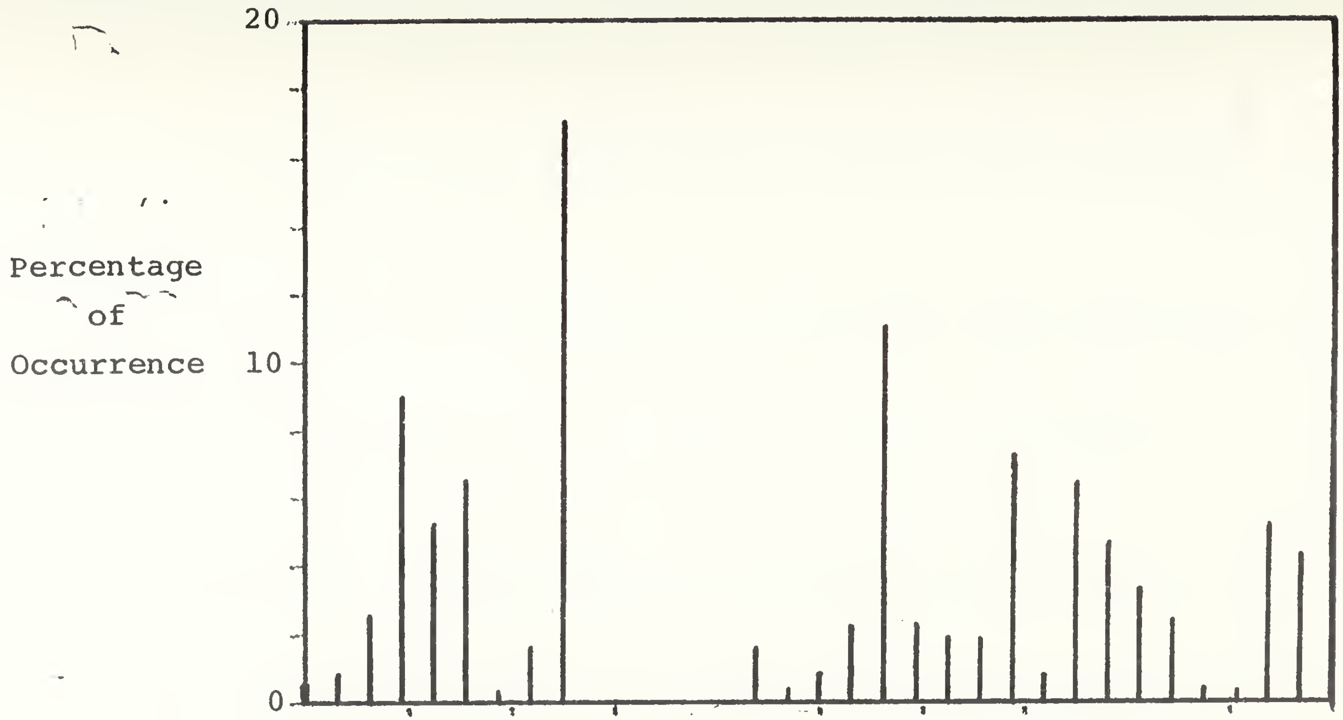


Figure 10. Simple substitution cipher
standard deviation = 3.81
Key = G

NUMBER OF OCCURRENCES

Character	KEY @	A	C	G	K	N
@	24	3	77	7	0	0
A	3	24	94	12	0	248
B	94	77	3	37	24	0
C	77	94	24	128	4	0
D	128	37	7	77	248	0
E	37	128	12	94	0	0
F	12	7	37	3	0	4
G	7	12	128	24	0	24
H	4	24	0	248	77	12
I	24	4	0	0	94	7
J	0	0	24	0	3	128
K	0	0	4	0	24	37
L	0	0	248	0	7	94
M	0	0	0	0	12	77
N	0	248	0	24	37	24
O	248	0	0	4	128	3
P	11	105	27	12	3	68
Q	105	11	27	32	3	93
R	27	27	105	160	76	33
S	37	27	11	33	63	48
T	33	160	12	27	93	3
U	160	33	32	27	68	3
V	32	12	160	105	48	63
W	12	32	33	11	33	76
X	63	76	3	93	27	32
Y	76	63	3	68	27	12
Z	3	3	76	48	105	33
[	3	3	63	33	11	160
/	33	48	93	3	12	27
]	48	33	68	3	32	27
^	68	93	48	76	160	11
--	93	68	33	63	33	105

Table No. V .- Simple substitution cipher
Table of number of occurren_
ces.

In this section, a digital polyalphabetic substitution very much alike to the Vigenere square, cited by Sinkov [Ref. 17], is designed. The originality of the scheme presented here is the fact that the different alphabets are used in a pseudorandom way and that this is generated through a simple algorithm in the computer.

The basis for the program to realize this cipher is provided by the same algorithm as for the simple substitution case, the only variation being that the key will change for each character to be ciphered. These changes of key are controlled by a program and thus the inverse transformation can be made to decipher by using the same program. This fact that we are using a different key each time is the same as using a new substitution alphabet for each character.

It must be set clear here that the key used was a single letter and not a number of letters equal to the message length. This single letter was used to initialize a register used as a counter. For each new letter of the message the register contents were increased by one each time until a specific number was reached, in which case the register was reset to zero. This specific number is the desired number of alphabets to be used. Figure 11 gives a graphical idea of how this was accomplished. In the figure, N represents the total number of alphabets to be used; it ranges from one, for a simple substitution, to 32 when using all the possible alphabets.

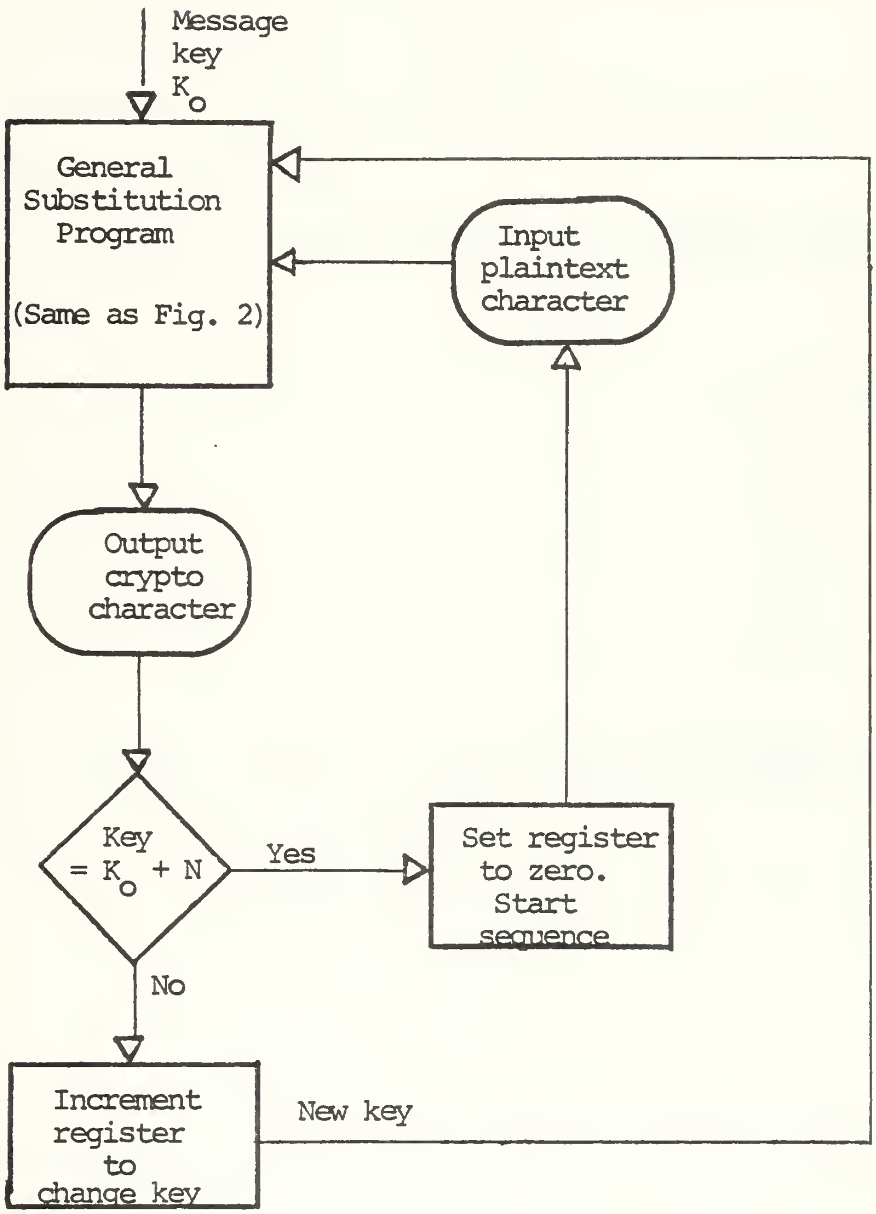


Figure 11. Psuedorandom cipher block diagram

The result expected for this cipher was the origination of an artificial language with 32 possible characters and with a letter frequency different than that of the plaintext message in natural English language.

To observe the results of this cipher two sets of transformations were made:

1. Using 15 alphabets and six different keys. The keys used were:

   a) @

   b) A

   c) C

   d) G

   e) K

   f) N

2. Using a single key and different number of alphabets, in the following order:

   a) 7 alphabets; key R

   b) 15 alphabets; key R

   c) 23 alphabets; key R

   d) 31 alphabets; key R

Figures 12 and 13 show some results obtained for the first set of transformations as a plot of percentage of occurrence of the 32 different characters. As can be observed, for the six cases, all the characters have a certain number of occurrences in the cryptogram obtained, thus giving rise to an artificial language of 32 characters witha quite different letter frequency than the plaintext of Figure 4.

In the same way, Figures 14 and 15 show some results obtained for the second set of transformations, which are essentially the same as the first set.

A measure of how different these results are from the plaintext is provided by the standard deviations in each case and are here listed to provide a means of evaluating the results achieved:

Number of alphabets	Key	Std. Deviation
15	@	1.528
15	A	1.528
15	C	1.528
15	G	1.528
15	K	1.528
15	N	1.528
7	R	1.467
15	R	1.545
23	R	1.407
31	R	1.329

These standard deviation values compared with the 3.81 for the plaintext, represent a significant flattening of the percentage of occurrence plots, or in other words, the cryptogram has a more equiprobable letter frequency.

A significant property of this scheme if we envision it as part of a digital communication system, is the fact that it offers no error propagation during the message processing.

The reason for this is the fact that each character is operated upon independently from all others. Thus, if there is an error in the bit representation of a letter, there will be an error in its transformation to crypto character or in the decryption of it and no error will occur in other characters due to it.

In the next section, a cryptographic scheme will be presented that although contributing to the communication system degradation, gives better results in the sense that a nearly equiprobable artificial language is achieved which represents a significant achievement for security of data transmission and/or data storage.

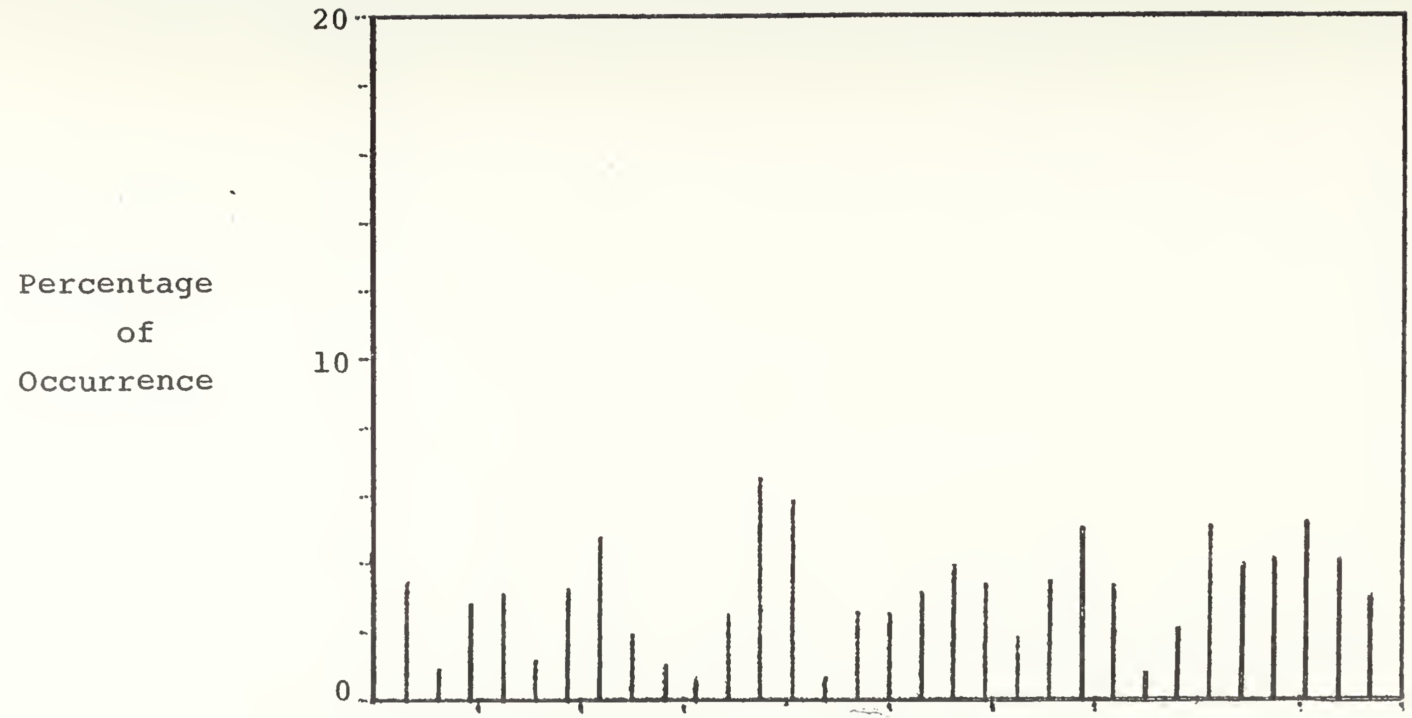


Figure 12. Pseudorandom cipher (Polyalphabetic substitution)
Standard deviation = 1.528
Number of alphabets: 15
Key = C

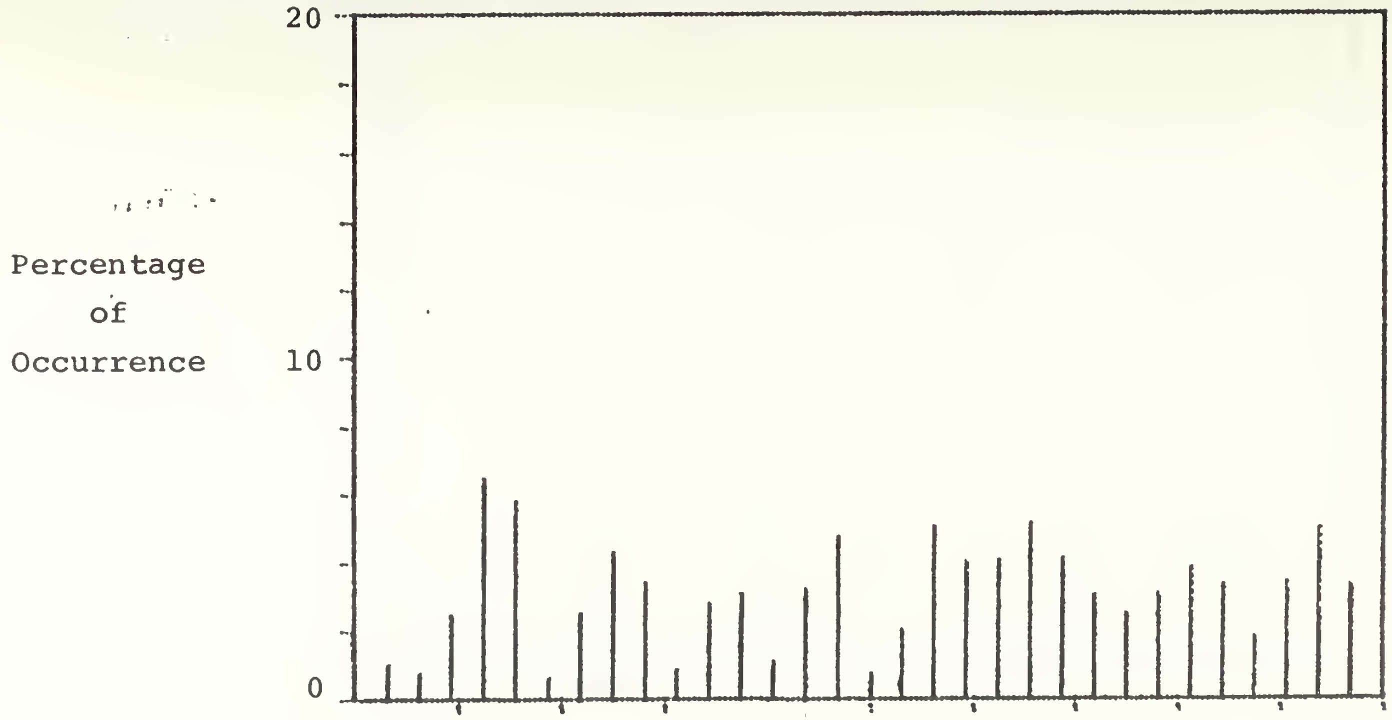


Figure 13. Pseudorandom cipher (Polyalphabetic substitution)
Standard deviation = 1.528
Number of alphabets: 15
Key = K

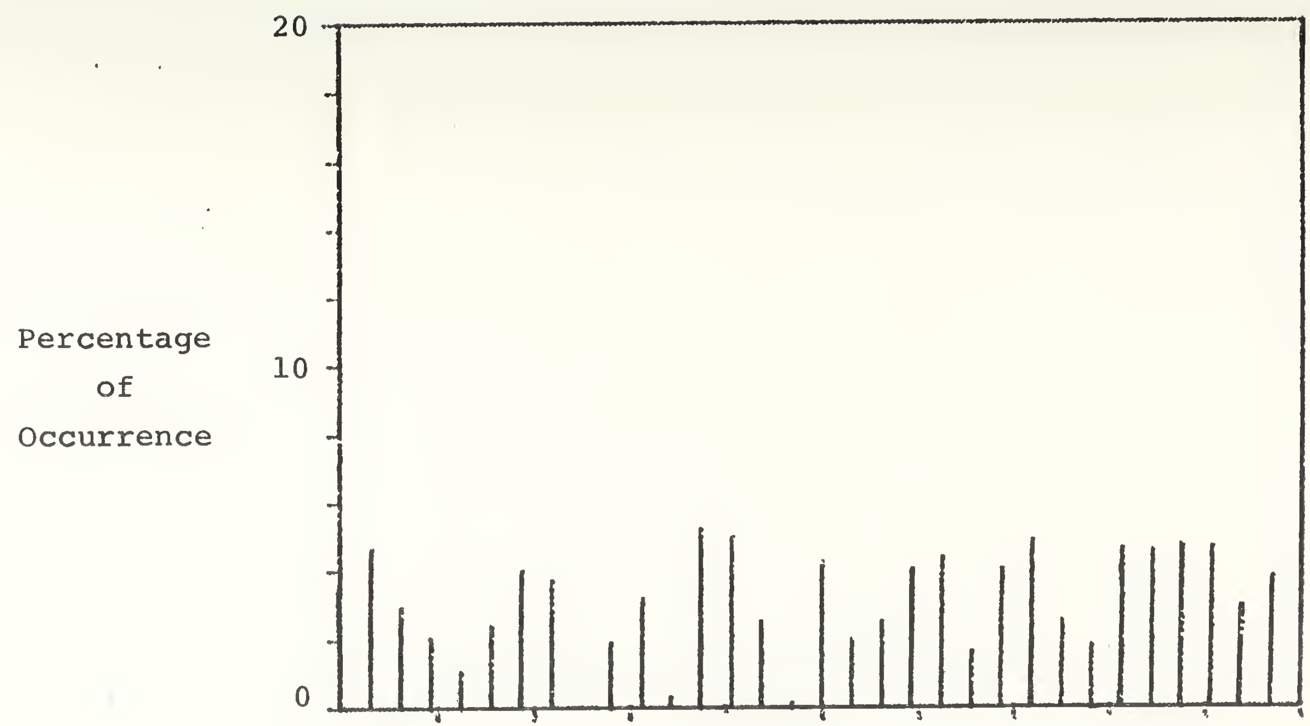


Figure 14. Pseudorandom cipher
Standard deviation = 1.467
Number of alphabets: 7
Key = R

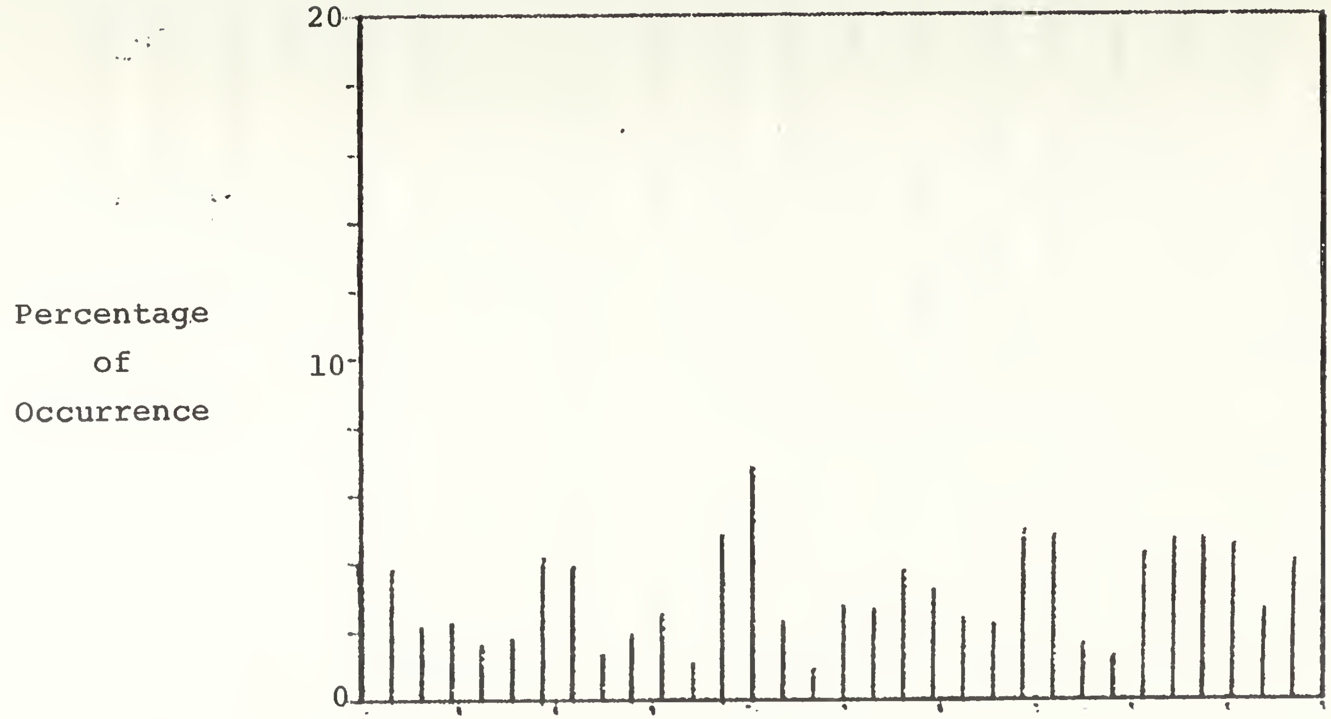


Figure 15. Pseudorandom cipher
Standard deviation = 1.407
Number of alphabets: 23
Key = R

VI. THE DATA-KEYED CIPHER

A. INTRODUCTION

In this section the data-keyed cipher is presented. First, a very general description of the system is given. Then the transfer function concept of the cipher and the reversibility and consistency of its is explained, together with the equated logical form of the transformation which the author appreciates as being a very meaningful representation of the cipher in logical form. After that the computer realization is presented in block diagram form. The test procedure for valuating secrecy accomplished and significant results are then given. Finally, the communication system degradation due to it is analyzed.

B. DESCRIPTION AND REALIZATION

Section IV explains how the PDP-11/40 computer is handled to realize the simple substitution cipher, consistency was shown with some examples and further, the known cryptoanalytic weakness of it was explained and graphically represented by Fig. 4 where it can be observed the frequency distribution of the plaintext and of some cryptograms and their similarity can be established.

The data-keyed cipher can be explained in a general form as the scrambling of the bits of a character by operating on them by past characters, either of the plaintext, when ciphering, or of the cryptogram, when deciphering.

Provided these past characters are far enough apart in the sequence their operation on the character to be transformed will result in a nearly random transformation. This idea was supported by the fact that for far enough distance between two letters in a written language there is nearly no statistical dependence between them.

Figure 16 provides the conceptual idea of this cipher. At this point, two significant characteristics that distinguish this cipher are to be emphasized:

1. From Figure 16(a) and (b) it can be seen that both diagrams can be conceived as a transfer function that essentially perform similar transformations on their inputs. An advantage is that when this is realized in the computer by a program, the same program will execute both transformations; that of ciphering and deciphering.

2. From Figure 16(b) it can be observed that there is no feedback present, that is, the outputs are not dependent on past outputs. The significance of this fact will be considered at the end of this section when system degradation for this cipher is treated.

The realization of this ciphering scheme again uses the basic transformations presented in Section IV, plus additional steps are included to accomplish the data-keyed function. The conceptual idea given in Figure 16 can now be expressed in logical equated form as:

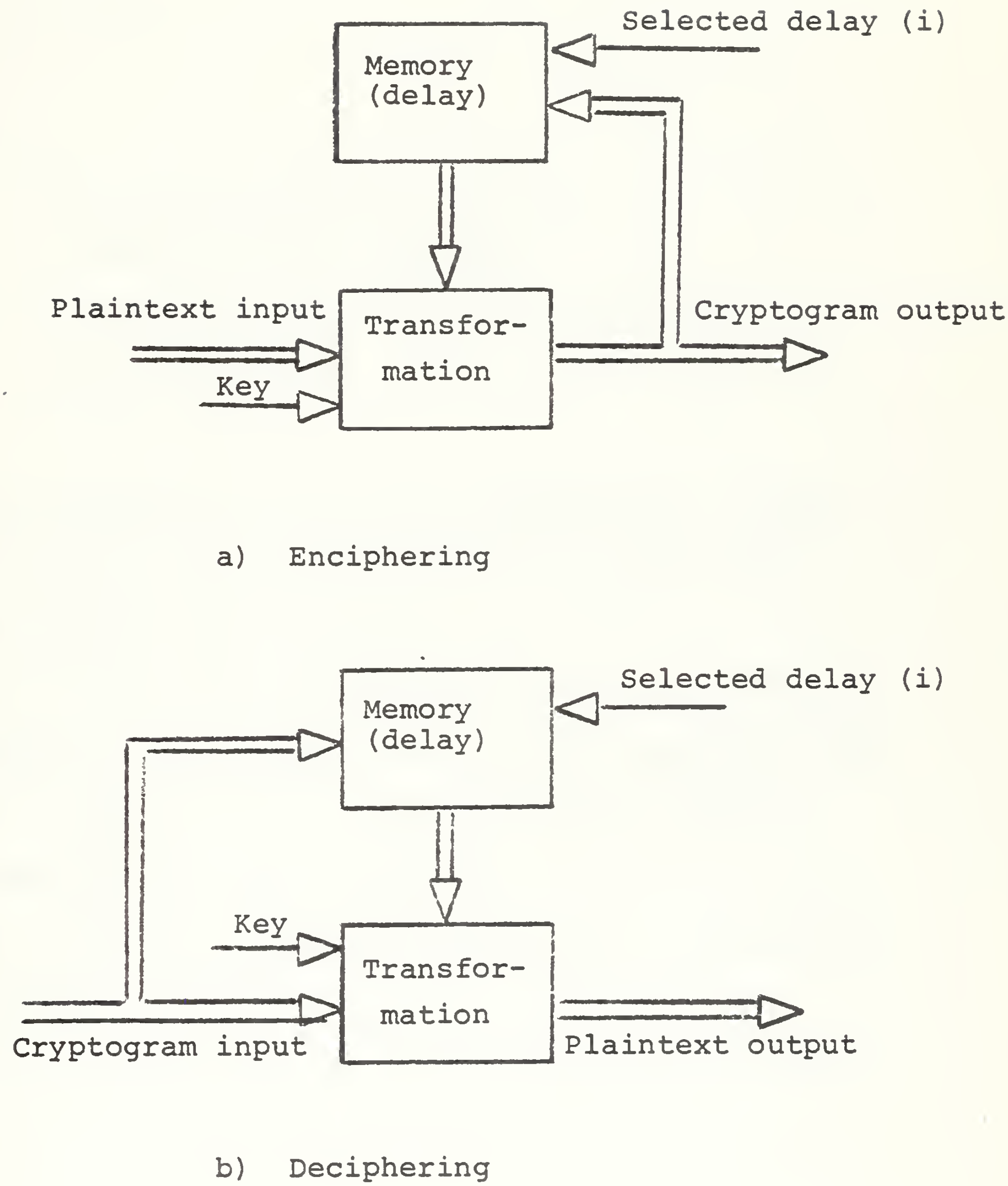


Figure 16. Data-Keyed Cipher-Concept

C I P H E R I N G : $C_j = (K + C_{j-1}) + P_j$

D E C I P H E R I N G : $P_j = (K + C_{j-1}) + C_j$

where

P_j = present plaintext character

C_j = present crypto character

C_{j-1} = "i" times preceding crypto character

K = key character

Again the operator used is the Exclusive-Or. These logical equations show the reversibility of the transformation and thus its consistency.

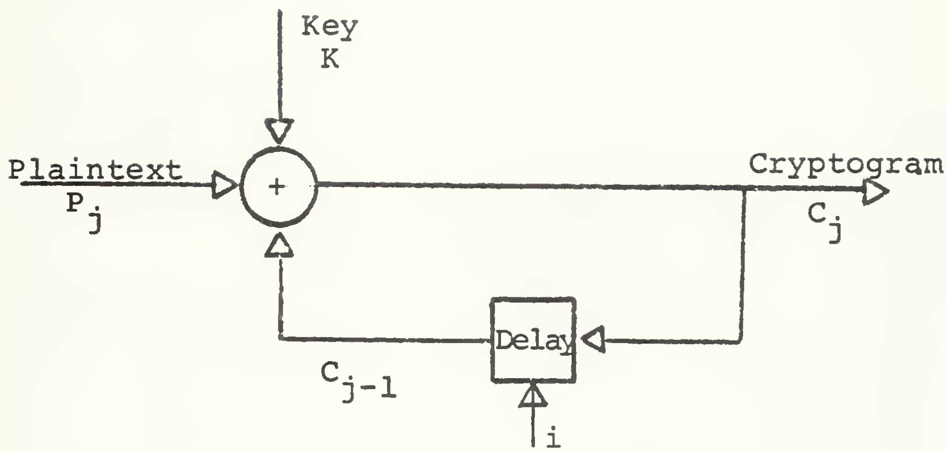
Figure 17 is now presented to give a more significant representation of the transformation to be realized. The index "i" is selective and it represents the distance between characters already explained.

Figure 18 shows the block diagram of the realization of this cipher in the PDP-11/40.

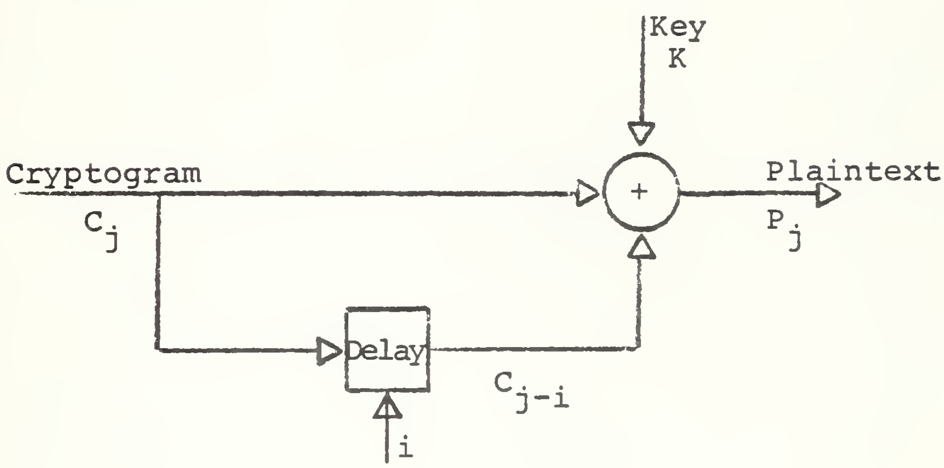
Appendix D gives the complete listing of the program used.

C. TEST PROCEDURE

The plaintext message used to test the results of this cipher scheme was the one presented in Section IV with its



a) Ciphering: $C_j = (K + C_{j-i}) + P_j$



b) Deciphering: $P_j = (K + C_{j-i}) + C_j$

Figure 17. Data-Keyed Cipher-Realization

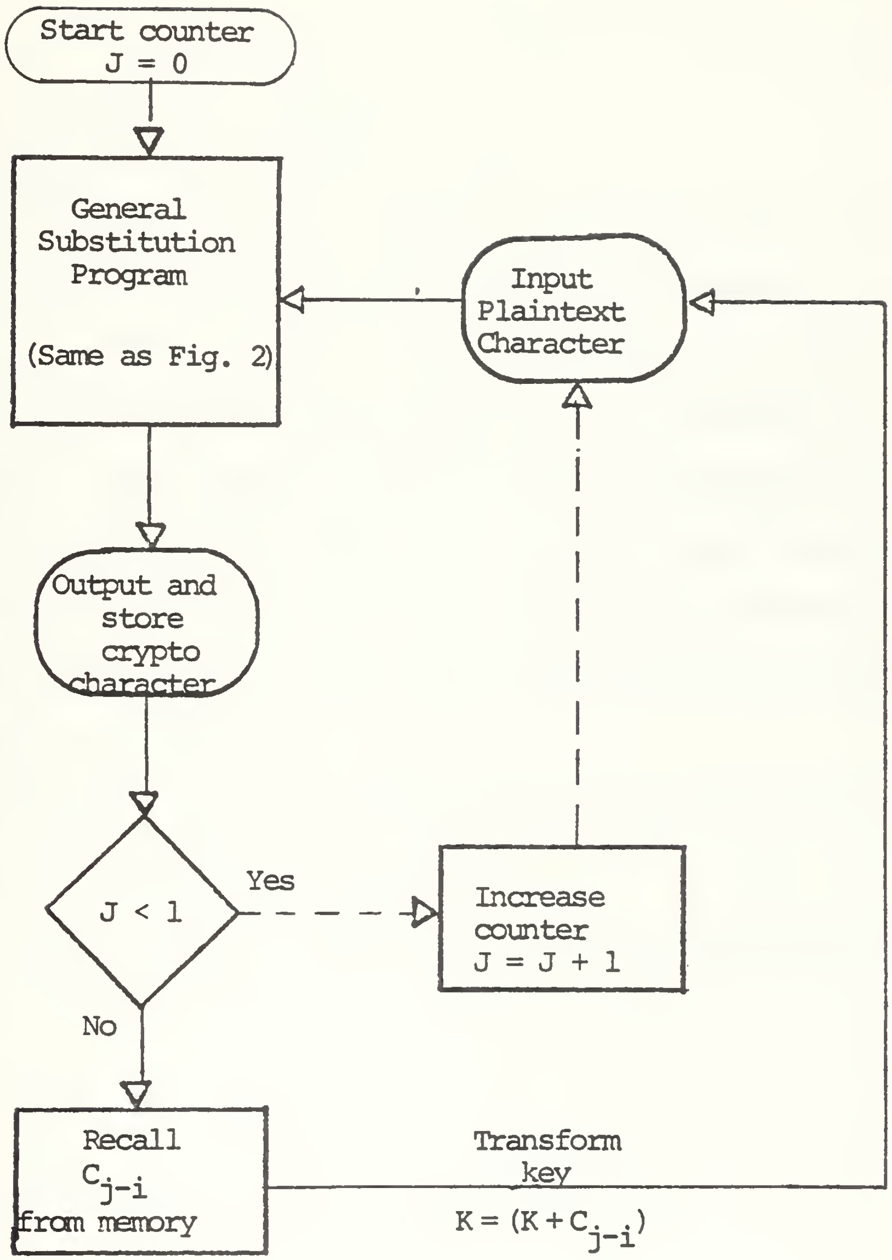


Figure 18. Data-Keyed Cipher-Block Diagram

statistics representative of the English language as shown in Figure 4.

This cipher, as depicted by Figure 17, has two possible choices of variables, namely:

- The key, with a total of 32.
- The delay factor "i" which could be varied from zero, for a simple substitution; up to any number n.

However, for any choice of n there will be the same amount of simple substitution characters at the beginning of the cryptogram. This disadvantage can be avoided by using for the first letters of the plaintext, meaningless text.

As for the simple substitution case, the intermediate keys were selected to reflect the transformations between sets C and D of Table II.

To observe the results obtained with this cipher two sets of transformations were made:

1. Using a fixed value of "i" and six different keys. For i = 7 and the keys:

   a) @

   b) A

   c) C

   d) G

   e) K

   f) N

2. For a fixed key and the following values of "i" (Key = J):

a) i = 2

b) i = 3

c) i = 10

d) i = 13

e) i = 17

f) i = 20

D. RESULTS

The results obtained for this cipher were, in all cases, significantly better than the Pseudorandom cipher of the previous section in the sense that the standard deviations were much lower, thus obtaining a nearly equiprobable text of cryptograms.

For the test procedure established, the following were the specific results obtained:

1. For a fixed value of "i" and using 6 out of 32 possible keys the following were the values of standard deviation obtained:

Key	"i"	Standard deviation
@	7	0.5783
A	7	0.6301
C	7	0.5395
G	7	0.5651
K	7	0.5608
N	7	0.6015

Figures 22 and 23 are some example plots for these cases. These figures are shown at the end of this section.

2. For a fixed key, different values of "i" were tried. The values of standard deviation obtained in each case were:

Key	"i"	Standard deviation
J	2	0.5761
J	3	0.5344
J	10	0.528
J	13	0.5317
J	17	0.4609
J	20	0.501

Figures 24 and 25 are some example plots for these cases and are presented at the end of this section.

We can now compare these results with the statistics of a plaintext English message with a standard deviation of 3.81 (see Figure 4). A significant flattening of the percentage of occurrence plots has occurred. In addition the statistical dependence of occurrence of the letter in the message has been hidden. The reason for this will be explained in the last part of this section where the nature of the ciphering scheme is explained in detail, together with the inherent degradation to a communication system due to it.

In Section IV it was stated, from Shannon [Ref. 15], that an ideal cipher may be an artificial language in which all letters are equiprobable and successive letters occurring independently. This is nearly the case for this cipher. Now a simple substitution, such as the one presented in Section V, can be performed on the message without making it easier to decipher.

3. A very meaningful characteristic of this scheme was the fact that the same program recovers or deciphers the message. Figures 19 and 20 present two examples of the encrypting results after being processed by the program corresponding to this cipher.

To give an idea of the number of occurrences of each character in the cryptograms for each of the 12 cases of (1) and (2), Tables VI and VII are next presented.

4. The implementation of this cipher in a digital computer can also be seen as the implementation of a code where the transformations are dependent on a key (a letter or character), the present letter to be encoded and some past crypto character.

E. COMMUNICATION SYSTEM DEGRADATION

Due to the nature of the process of ciphering and deciphering of this system, it can be said that when it comes to play an integral part of a communication system, it, at the most, will double the probability of block error. Here the block length has been 8 bits corresponding to a

```
CTHIS_BOOK_IS_DESIGNED_PRIMARILY_FOR_USE_AS_A_FIRST_YEAR
_GRADUATE_TEXT_IN_INFORMATION_THEORY_SUITABLE_FOR_BOTH_E
NGINEERS_AND_MATHEMATICIANS_@_IT_IS_ASSUMED_THAT_THE_REA
DER_HAS_SOME_UNDERSTANDING_OF_FRESHMAN_CALCULUS_AND_ELEM
ENTARY_PROBABILITY_AND_IN_THE_LATER_CHAPTERS_SOME_INTROD
UCTORY_RANDOM_PROCESS_THEORY_@_UNFORTUNATELY_THERE_IS_ON
E_MORE_REQUIREMENT_THAT_IS_HARDER_TO_MEET_@_THE_READER_M
UST_HAVE_A_REASONABLE_LEVEL_OF_MATHEMATICAL_MATURITY
```

a) Plaintext message (input)

```
CG[Z@LQ\\_GP@@VZLUS]VWJIM_]_G]UCAZQNKKEUMXABYGP_LHFNCQRN
@\G\TGPIV@@ZO@LS[LZWZ\M]YKP[W@JV_WAAK@L\HUP^]LYPIIQPJWEV
T][_\QDFH_VX@_VASYXZGUV[QTHFTI\LM^HJVILJSX_FAR^MOOTPMCX_
XYELVQHCHU[ZJGUT^DKMXZR^SPGQ]FWOU@\__[K_W_\IPMKCUP[EYR]]
S_LWGHANBS]UVR^TEIAW[UB^XEF\]I]\OSG@]R_OHUF@ABSQ^I\]FC_V
HYKQGICGZT\]YE@FVTZMIIG]PXKGEZKK]]WFBLVYJ[XLNK]_K]QVNGQR
]AZZOQMC[CLPNWSUVD@GUUDI\DL\WDS_MHKPKZUYJDXLLQSEKRZ[Z@O[
MBMGQRJ]APAFWPJQ\RPYQLUWIT_E]YIY[CTSS[NS[Q[O]YIEJ[LE
```

b) Cryptogram message (output)

Figure 19. Data-Keyed cipher
Encrypting process

```
DTHIS_BOOK_IS_DESIGNED_PRIMARILY_FOR_USE_AS_A_FIRST_YEAR
_GRADUATE_TEXT_IN_INFORMATION_THEORY_SUITABLE_FOR_BOTH_E
NGINEERS_AND_MATHEMATICIANS_@_IT_IS_ASSUMED_THAT_THE_REA
DER_HAS_SOME_UNDERSTANDING_OF_FRESHMAN_CALCULUS_AND_ELEM
ENTARY_PROBABILITY_AND_IN_THE_LATER_CHAPTERS_SOME_INTROD
UCTORY_RANDOM_PROCESS_THEORY_@_UNFORTUNATELY_THERE_IS_ON
E_MORE_REQUIREMENT_THAT_IS_HARDER_TO_MEET_@_THE_READER_M
UST_HAVE_A_REASONABLE_LEVEL_OF_MATHEMATICAL_MATURITY
```

a) Plaintext message (input)

```
D@\]GKV[[_GP@@VZLRTZQPMNJ_]_G]UCA]VILLBRJXABYGP_LOAIDVUI
G\G\TGPIVGG]HGKT\LZNZ\M]YLW\PGMQXWAAK@L\HRWYZK^WNIQRJWEV
TZ\X[VCAO_VX@_VAS^_]@RQ\VTHFTI\LMYOMQNKMTX_FAR^MOHSWJD_X
_YELVQHCHR\]M@RSYDKMXZR^SW@VZAPHR@\__[K_WX[NWJLDRR[EVR]]
SXKP@OFIES]UVR^TENFP\REY_EF\]I]\OT@GZUXHOUF@ABSQ^N[ZADXQ
OYKQGICGZS[Z^BGAQTZMIIG]P_L@B]LLZ]WFBLVYJ\_KILZXL]DVNGQR
]F]]HVJD\CLPNWSUVCG@RRCN[DL\WDS_MOLWL]R^MDXLLQSEKU]\]GH\
JBMGQRJ]AWFAPWMV[RPYQLUWISXBZ^N^\CTSS[NS[V\HZ^NBM[LE
```

b) Cryptogram message (output)

Figure 20. Data-Keyed cipher
Encrypting process

NUMBER OF OCCURRENCES

	KEY (i = 7)					
Character	@	A	C	G	K	N
@	36	33	40	46	45	32
A	35	38	37	40	40	34
B	35	40	33	32	32	42
C	55	50	51	52	53	50
D	47	42	56	50	42	48
E	46	51	52	49	46	59
F	47	55	41	42	44	47
G	50	42	41	40	46	36
H	41	35	48	35	43	41
I	38	44	44	38	41	52
J	34	41	28	31	29	33
K	47	40	40	44	38	34
L	44	37	34	47	48	37
M	42	49	39	45	45	47
N	29	29	32	29	29	33
O	32	32	42	38	37	33
P	51	37	47	38	36	45
Q	43	57	48	44	48	51
R	50	55	45	43	61	58
S	58	53	62	60	61	50
T	53	39	42	51	49	46
U	40	54	43	47	50	41
V	51	51	48	50	52	63
W	38	38	49	51	50	53
X	59	62	45	54	56	47
Y	64	61	53	56	53	49
Z	43	37	54	40	38	50
[	37	43	51	51	52	36
/	52	40	46	37	39	43
]	51	63	58	55	51	60
^	52	52	46	60	42	58
_	52	52	57	57	56	44

Table No. VI .- Data-keyed cipher Table of number of occurrences.

NUMBER OF OCCURRENCES

Character	" i " VALUES (KEY = J)					
	2	3	10	13	17	20
@	37	42	40	32	42	46
A	41	40	36	35	41	48
B	48	39	49	34	36	40
C	44	37	38	40	29	39
D	34	43	47	41	41	50
E	43	41	46	49	50	47
F	47	43	35	47	42	48
G	45	46	48	39	40	33
H	48	39	44	33	48	38
I	38	36	34	53	45	35
J	32	54	36	46	42	38
K	52	42	40	38	41	31
L	41	42	37	38	40	38
M	37	41	34	44	36	36
N	45	28	52	48	35	42
O	26	45	42	41	50	49
P	44	46	49	59	51	50
Q	36	52	58	50	48	45
R	61	36	46	53	47	45
S	46	65	37	56	48	62
T	60	62	43	43	52	48
U	49	50	47	54	56	50
V	54	44	45	55	40	55
W	46	50	62	38	50	49
X	43	58	53	36	46	44
Y	49	42	51	49	49	52
Z	44	45	41	49	57	54
[	44	57	53	55	49	36
/	60	50	48	40	39	45
]	54	42	62	46	55	55
^	52	50	55	55	53	56
_	52	45	44	56	54	48

Table No. VII.- Data-keyed cipher
Table of number of
occurrences.

byte. It must be emphasized that, although for ease of computer realization the 8-bit byte was used to represent a letter; only 5 bits could have been enough since we are using only 32 letters or characters.

This increase in probability of error can be said to be significant but with the availability of error correcting codes the initial probability of error can be reduced as desired and appropriately so that doubling it when using the cryptosystem will not be that significant. Further, since a computer is being used to implement it, it also can be used to realize a suitable error correcting scheme. In the next section, a suitable error correcting scheme is presented, that will essentially overcome this degradation.

The examples that follow are intended to explain how the probability of block error is doubled and also the existence of a transient simple substitution for the first "i" characters.

Based on these two examples the following observations can be made:

1. There is a transient simple substitution for the first "i" characters when enciphering. This is the case of C_1, C_2 and C_3 from Example 1.

2. After the transient simple substitution, the crypto characters are a result of a number of plaintext characters. And, the higher the index of the crypto to be obtained, the more the number of plaintext characters on which it depends.

Example No. 1 Enciphering process

Transformation: $C_j = (K + C_{j-i}) + P_j$

Plaintext sequence: $P_1, P_2, P_3, P_4, P_5, P_6, P_7, P_8, P_9$

Let $i = 3$

$C_1 = K + P_1$

$C_2 = K + P_2$

$C_3 = K + P_3$

$C_4 = K + C_1 + P_4 = K + (K + P_1) + P_4 = P_1 + P_4$

$C_5 = K + C_2 + P_5 = K + (K + P_2) + P_5 = P_2 + P_5$

$C_6 = K + C_3 + P_6 = K + (K + P_3) + P_6 = P_3 + P_6$

$C_7 = K + C_4 + P_7 = K + (P_1 + P_4) + P_7$

$C_8 = K + C_5 + P_8 = K + (P_2 + P_5) + P_8$

$C_9 = K + C_6 + P_9 = K + (P_3 + P_6) + P_9$

$C_{10} = K + C_7 + P_{10} = P_1 + P_4 + P_7 + P_{10}$

$C_{11} = K + C_8 + P_{11} = P_2 + P_5 + P_8 + P_{11}$

$C_{12} = K + C_9 + P_{12} = P_3 + P_6 + P_9 + P_{12}$

$C_{13} = K + C_{10} + P_{13} = K + P_1 + P_4 + P_7 + P_{10} + P_{13}$

. . .

<u>Example No. 2</u> Deciphering process

Transformation: $P_j = (K + C_{j-i}) + C_j$

Cryptogram sequence: $C_1, C_2, C_3, C_4, C_5, C_6, C_7, C_8, C_9$

Let i = 3, as before

$P_1 = K + C_1$

$P_2 = K + C_2$

$P_3 = K + C_3$

$P_4 = K + C_4 + C_1$

$P_5 = K + C_5 + C_2$

$P_6 = K + C_6 + C_3$

$P_7 = K + C_7 + C_4$

$P_8 = K + C_8 + C_5$

. . .

$P_n = K + C_n + C_{n-i}$

3. The order of dependency observed in Example 1 is different for the deciphering case, where the recovering of the text is just dependent on two crypto characters. Thus, one error in the crypto sequence will just give rise to two errors in the plaintext.

Figure 21 gives an example of the transient simple substitution explained. The value of "i" chosen there is 50. As an example it can be observed here that for the first 50 characters of the plaintext the letter R is always substituted by the letter C.

```
ATHIS_IS_AN_EXAMPLE_OF_A_CYCLIC_ERROR_CORRECTING_CODE_AP
PLIED_TO_A_CIPHERED_MESSAGE___NOISE_GENERATED_IN_A_PROGR
AM_IS_MODULO_TWO_ADDED_TO_THE_MESSAGE_TO_TEST_THE_EFFECT
IVENESS_OF_THE_CODE@
```

a) Plaintext

Transient substitution

```
AEYXBNXBNP_NTIP\A]TN^WNPNRHR]XRNTCC^CNR^CCTREX_VNR^UQGXC
OUZZUAKZGPBSUEWZDZU@^\@OXTZJMMQ]V@ZMUP]QK_CZW@]^IXMNFT\F
QWD_BL_[PCWZD@FPBPZ]P_DHSOH_TDQQBOQ][MOYNHZKMCK_PBRVPP]G
TH__WEHJ^QNGY^CRQQ\R
```

b) Cryptogram

Figure 21. Data-keyed Cipher - Example of transient substitution. i = 50.

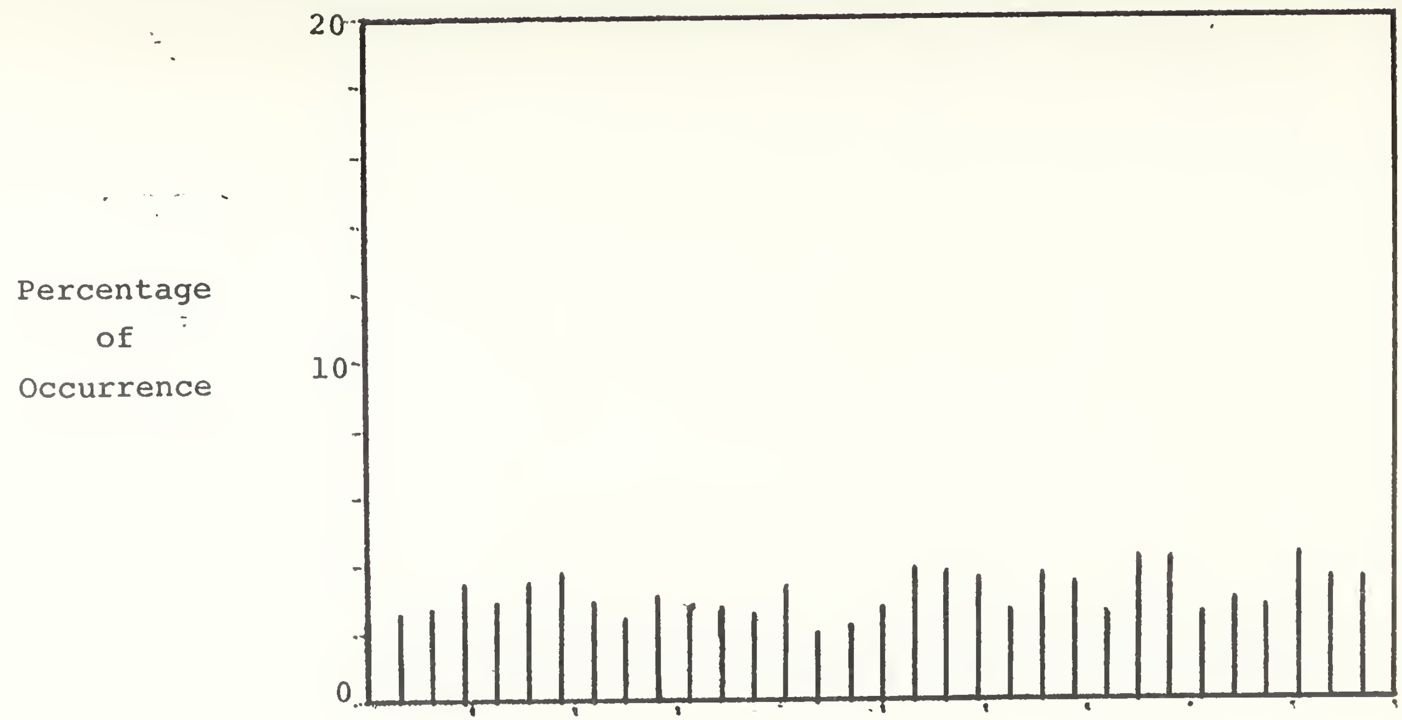


Figure 22. Data-keyed cipher
Standard deviation = 0.6301
Key = A i = 7

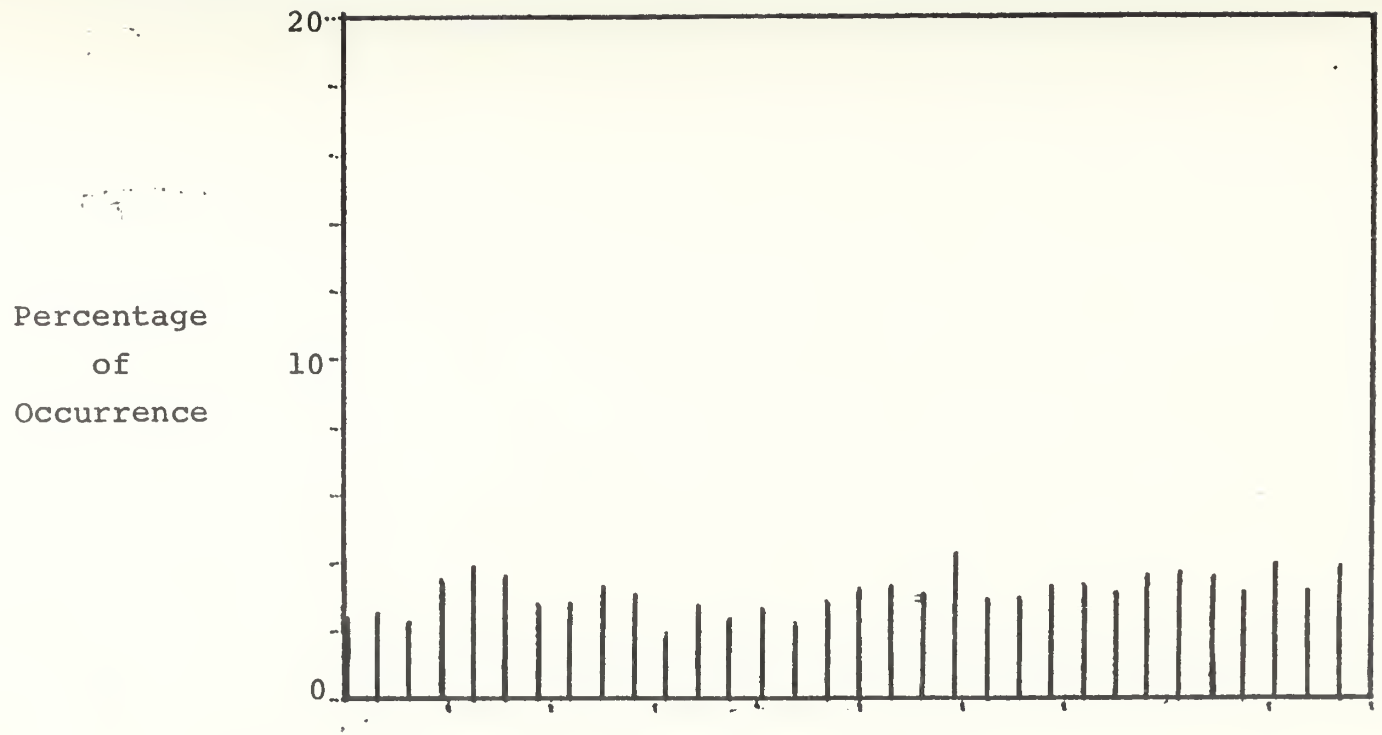


Figure 23. Data-keyed cipher
Standard deviation = 0.5395
Key = C i = 7

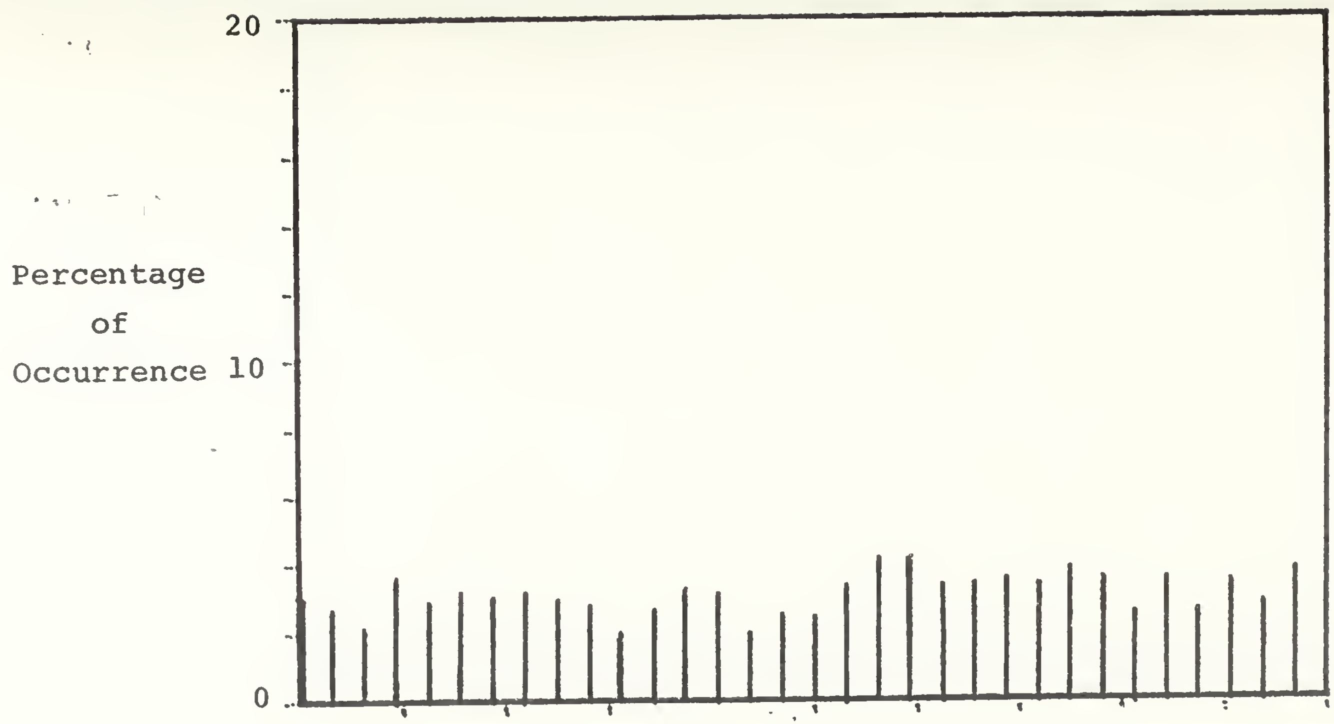


Figure 24. Data-keyed cipher
Standard deviation = 0.501
Key = J i = 2

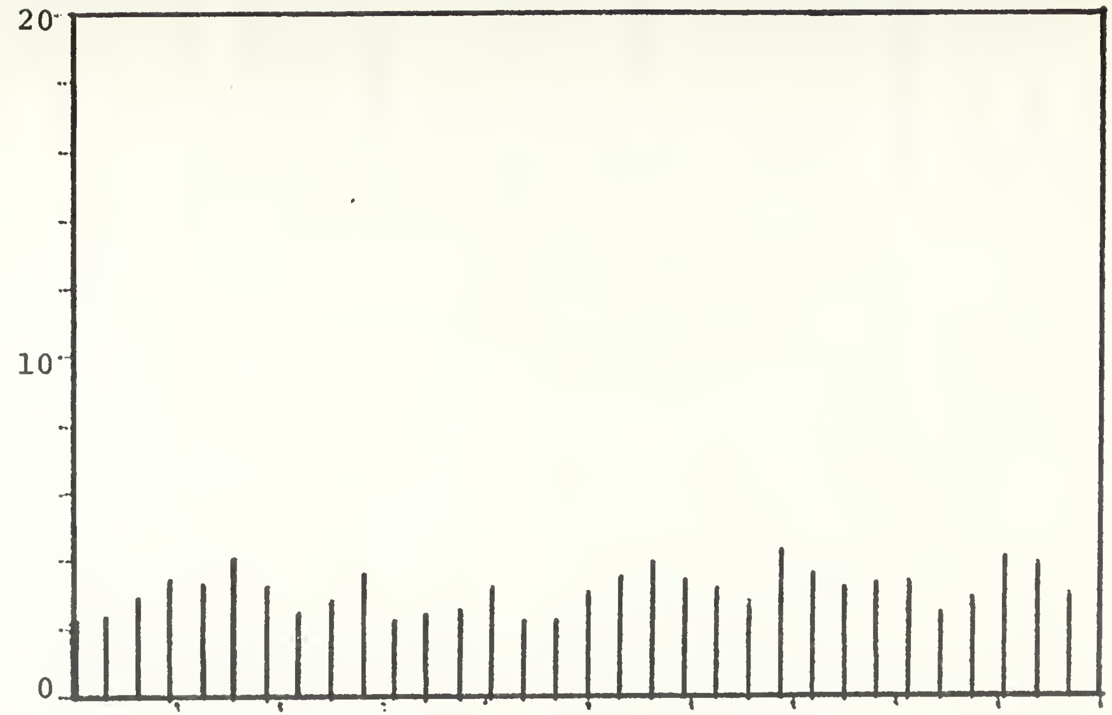


Figure 25. Data-keyed cipher
Standard deviation = 0.4609
Key = J i = 17

VII. ERROR CORRECTING SCHEME

The data-keyed cipher of the last section offers to the system a degradation in the sense that the probability of word error is doubled due to the nature of the encipherment process, as was explained. This increase in error will undoubtedly affect the legibility of any message. Thus it was necessary to look into error correcting codes that will eventually overcome this present disadvantage. Again the availability of the digital computer proved to be very useful for enciphering the message and to encode it for transmission.

The error correcting code developed was intended for transmission over a memoryless binary symmetric channel. A memoryless channel is the one on which noise does not depend upon previous events. A binary symmetric channel is one for which the probability of a zero to be changed to a one, is equal to the probability of a one to be changed to a zero, during transmission.

Notation that will encountered through this section follows:

k = Number of information digits

m = Number of check bits

n = Code word length ($n = k + m$)

e = Maximum number of correctible bit errors in one word

R = Data rate ($R = k/n$)

β = Binary symmetric channel parameter $p(1/0) = p(0/1)$

d = Hamming distance between code words.

A. BEST CODE DETERMINATION

The noise channel theorem as stated by Shannon [Ref. 14] is:

> Let a discrete channel have the capacity C bits/sec. and a discrete source has the entropy per second H. If $H < C$ there exists a coding scheme such that the output of the source can be transmitted over the channel with an arbitrarily small frequency of errors. If $H > C$, it is possible to encode the source so that the equivocation is less than $H - C + \varepsilon$, where ε is arbitrarily small. There is no method of encoding that gives an equivocation less than $H - C$.

The discrete source entropy for long messages consistin of discrete symbols is given by

$$H(x) = - \sum_{i=1}^{n} p_i \log p_i$$

where p_i is the probability of occurrence of a given symbol. In the situation where the symbols are transmitted over a noisy channel a given symbol x_i may be received as y_i. Shannon's measure of uncertainty at the receiver of what was actually transmitted is defined as:

$$H(x/y) = - \sum_{x} \sum_{y} p(x_i, y_i) \log p(x_i/y_i)$$

For the binary symmetric channel this uncertainty is given by:

$$H(x/y) = -(\beta \log \beta + (1-\beta) \log (1-\beta))$$

Then the channel capacity is given by

$$C = H(x) - H(x/y) \quad \text{maximized for } H(x) .$$

A significant parameter commonly used is the probability of word error in the message instead of the uncertainty measure. The probability of word error is defined as:

$$P(e) = \frac{\text{Number of wrong decoded words}}{\text{Number of words in message}}$$

It must be noted at this point that there will not necessarily be a code word for each ASCII character used. In fact this was the case for the code implemented, where each 4 bits of the message sequence is encoded into a 15-bit word. Thus, each 8-bit ASCII character was encoded into two words for transmission.

A "best code" means one that has least probability of error for any give channel β and the highest rate given by the ratio of information bits over the bit-length of each code word. The error correction ability of the code can be derived from the Varsharmov-Gilbert-Sacks condition (upper bound)

$$2^m > \sum_{i=0}^{2e-1} \binom{n-1}{i}$$

which is a sufficient but not necessary condition. And from the Hammings lower bound inequality

$$2^m \geq \sum_{i=0}^{e} \binom{n}{i}$$

which is a necessary but not sufficient condition for designing an e-tuple error correcting code.

Conversely, using these conditions, once a code is chosen and specified by its rate (R) and code word length (n), the number of correctible e-tuples can be determined.

The theoretical value of probability of error is given by Ash [Ref. 18]:

$$p(e) = 1 = \sum_{i=0}^{e} N_i \, \beta^i (1-\beta)^{n-i}$$

where N_i is the number of correctible e-tuple errors, and $e_i = 0,1,2,\ldots,$ up to the maximum number of correctible errors per word.

The Hamming distance (d) is the minimum distance between code words. If d happens to be even and the maximum value of e is given by $(d-1)/2$, this will yield a fraction.

Then the number of maximum e-tuple errors is given by Shiva [Ref. 19]

$$\frac{\text{Number of correctible } d/2 \text{ errors}}{\text{Total number of } d/2 \text{ errors}} = 1 - \frac{\frac{\mu(\mu+1)}{2}}{\genfrac{}{}{0pt}{}{n}{d/2}}$$

where $$\mu = \frac{d!}{(\frac{d}{2})!(\frac{d}{2})!}$$

For the same channel (β constant), reducing the probability of error results in a reduction of the code rate. Working backwards, for any given probability of error and word length, one can estimate the information length and code rate by using the Varsharmov-Gilbert-Sacks condition.

In the present work a cyclic code with a rate $R = 4/15$ is implemented to overcome the degradation due to the noisy channel. Its effectiveness was tested by simulating transmission over a binary symmetric channel with different values of β.

B. THE (15,4) CYCLIC CODE AND ITS COMPUTER REALIZATION

The theory of Cyclic Codes and their representation by means of a k-stage feedback shift register is very well treated by Ash [Ref. 18].

1. Selection of Polynomial

In order to be compatible with the 16-bit organization of the PDP-11/40, the characteristic polynomial for

this code was chosen from Appendix C of Peterson [Ref. 20], and it was

$$G(x) = x^4 + x + 1$$

which is an irreducible polynomial and which can be represented by a 4-stage shift register as shown in Figure 26. Since G(x) is a maximum period irreducible polynomial, with a period $2^4 - 1 = 15$, it divides the polynomial $x^{15} + 1$ (modulo 2). Thus, the check polynomial for this code will be

$$H(x) = \frac{x^{15} + 1}{G(x)} = x^{11} + x^8 + x^7 + x^5 + x^3 + x^2 + x + 1$$

The polynomial chosen originates a (15,4) cyclic code, that is, a code where

$$k = 4$$

$$m = 11$$

$$n = 15$$

The coefficients of the check polynomial for the code word 000100110101111. Since the code is cyclic, any cyclic shift of the check word and any linear combination of code words is another code word. This property of the cyclic code represents an advantage for decoding purposes.

Procedure:

1. The 4-bit word to be coded is loaded in parallel into the 4-stage shift register feedback configuration.
2. Then the shift register is let to run until a 15-bit serial output (the code word) is obtained.

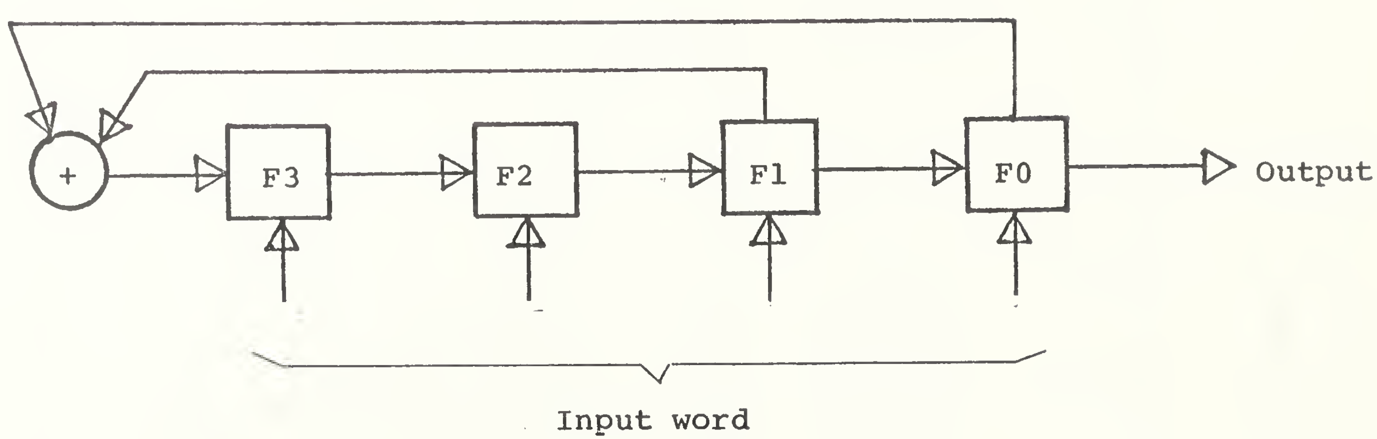


Figure 26. 4-stage encoder of the characteristic polynomial $G(x) = x^4 + x + 1$

2. Computer Realization of Encoder

Encoding in a digital computer is accomplished by realizing the shift-register operations by implementing a matrix multiplication of the message word by a generator matrix.

The generator matrix for the characteristic polynomial $G(x) = x^4 + x + 1$ used, was

$$[G]_{4,15} = \begin{bmatrix} 1&0&0&0&1&0&0&1&1&0&1&0&1&1&1 \\ 0&1&0&0&1&1&0&1&0&1&1&1&1&0&0 \\ 0&0&1&0&0&1&1&0&1&0&1&1&1&1&0 \\ 0&0&0&1&0&0&1&1&0&1&0&1&1&1&1 \end{bmatrix}$$

which when multiplied by the message word $[x]_{1,4}$, yielded the code word $[w]_{1,15}$.

A further comment can be made on the structure of the generator matrix: The four rows are code words and they are linearly independent, and, any of the other code words can be obtained by linear combination of these four rows. For ease of computer implementation, to obtain a code word it was only needed to exclusive-or the rows of $[G]_{1,15}$ where a 1 occurs in the message word. For example,

$$[X]_{1,4} = 1\ 1\ 0\ 0 \qquad \text{(message word)}$$

First row of G =	1 0 0 0 1 0 0 1 1 0 1 0 1 1 1 +
Second row of G =	0 1 0 0 1 1 0 1 0 1 1 1 1 0 0
Code word	1 1 0 0 0 1 0 0 1 1 0 1 0 1 1

Appendix E shows the complete listing of this encoding program.

3. <u>Minimum Distance Decoder</u>

Table VIII gives the code words for the 16 possible message words when the (15,4) cyclic code is used. It can be observed that the Hamming distance between these code words is 8. That is, the number of different digits between code words is 8 ($d = 8$).

With the minimum distance decoder, if any combination of $\frac{d-1}{2}$ or less errors occur in a received code word, it can be corrected with absolute certainty. For this code, any 3 or less errors can be corrected successfully.

For the case when 4-digit errors occur ($e = 4$), the Varsharmov-Gilbert-Sacks condition (Upper bound)

$$2^m \quad \sum_{i=0}^{2e-1} \binom{n-1}{i}$$

is not satisfied and thus there exists an uncertainty on whether a 4-digit error will be corrected. It has been found experimentally that 67.8% of different combinations of 4-digit errors can be corrected. Appendix G shows the complete listing of the decoding program.

C. NOISY CHANNEL SIMULATION

Table IX provides the expected probabilities of error for transmission over a noisy binary symmetric channel when using the (15,4) cyclic code presented, as given by

Information Word	Coded Word
0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 1	0 0 0 1 0 0 1 1 0 1 0 1 1 1 1
0 0 1 0	0 0 1 0 0 1 1 0 1 0 1 1 1 1 0
0 0 1 1	0 0 1 1 0 1 0 1 1 1 1 0 0 0 1
0 1 0 0	0 1 0 0 1 1 0 1 0 1 1 1 1 0 0
0 1 0 1	0 1 0 1 1 1 1 0 0 0 1 0 0 1 1
0 1 1 0	0 1 1 0 1 0 1 1 1 1 0 0 0 1 0
0 1 1 1	0 1 1 1 1 0 0 0 1 0 0 1 1 0 1
1 0 0 0	1 0 0 0 1 0 0 1 1 0 1 0 1 1 1
1 0 0 1	1 0 0 1 1 0 1 0 1 1 1 1 0 0 0
1 0 1 0	1 0 1 0 1 1 1 1 0 0 0 1 0 0 1
1 0 1 1	1 0 1 1 1 1 0 0 0 1 0 0 1 1 0
1 1 0 0	1 1 0 0 0 1 0 0 1 1 0 1 0 1 1
1 1 0 1	1 1 0 1 0 1 1 1 1 0 0 0 1 0 0
1 1 1 0	1 1 1 0 0 0 1 0 0 1 1 0 1 0 1
1 1 1 1	1 1 1 1 0 0 0 1 0 0 1 1 0 1 0

TABLE VIII. Message words and their correspondent code word for the (15,4) cyclic code

Channel β	Probability of error P(e)
0.07050	5.4480×10^{-3}
0.09797	2.9176×10^{-2}
0.12426	6.2425×10^{-2}
0.13992	1.2542×10^{-1}
0.1709	1.8780×10^{-1}
0.26613	4.9052×10^{-1}

TABLE IX. P(e) vs. channel β for the code (15,4)

Cetinyilmaz [Ref. 21]. In the same reference a noise generating program is presented to simulate different conditional probabilities of error for the BSC. The same program was used in this thesis to simulate a noise BSC and to test the effectiveness of the code implemented. Appendix F gives a listing of the program.

Having the enciphering scheme, the error correcting code and a mean for introducing noise into the message to reflect different values of β for the channel, all were combined to simulate a Secure Digital Communication System, as depicted by Figure 27.

The following is the complete program flow for the system:

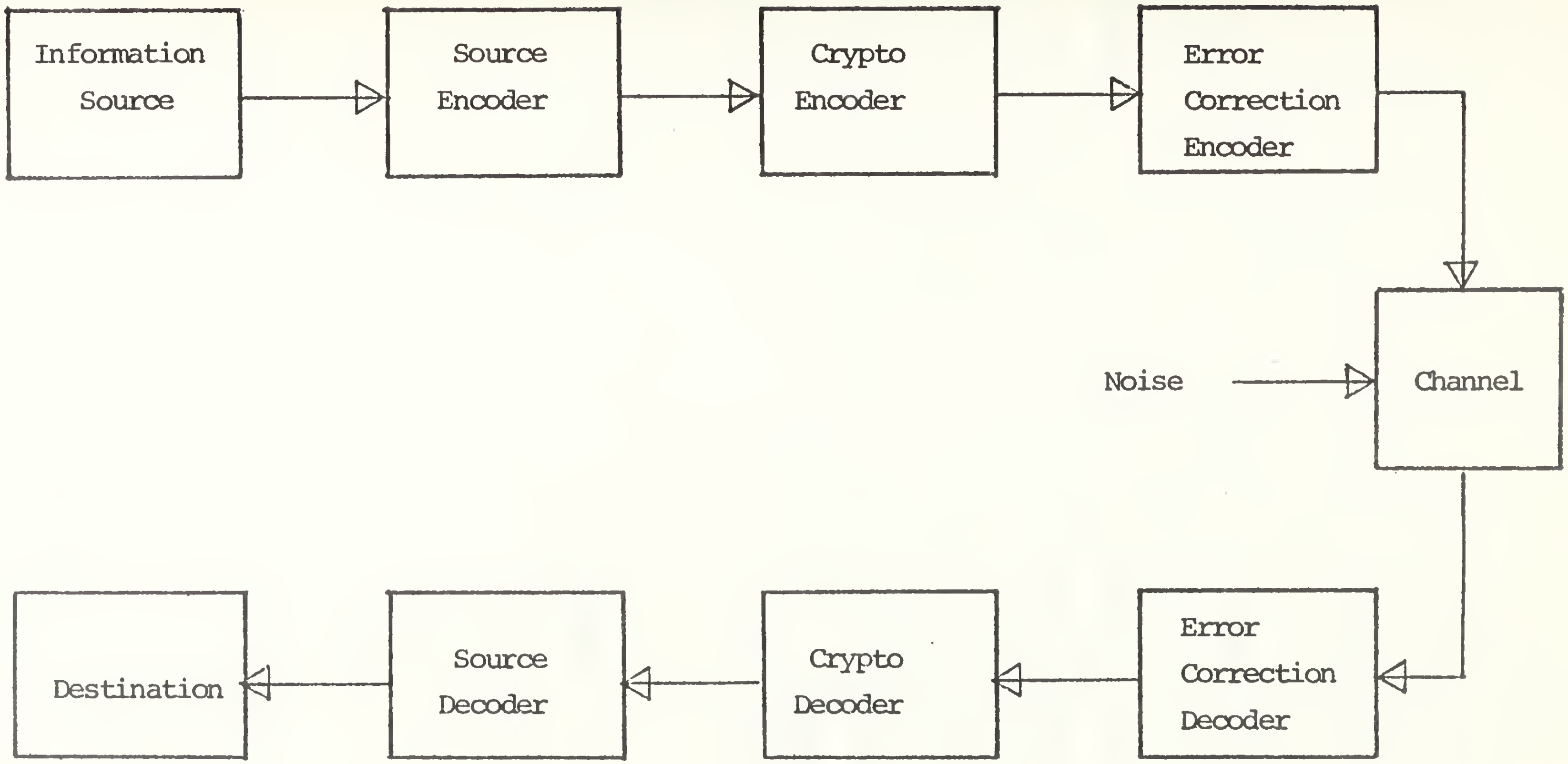


Figure 27. Secure digital communication system block diagram

a) Input program (address 20000 to 20036) - The message is typed in. The program stores the message in ASCII code form into memory locations 30002-32000 (16-bit form).

b) Data-keyed cipher program (10000-11044) - The key to be used is typed in, the program stores it at 30000. The program takes the message from 30000-32000, ciphers it and then stores it at 40000-42000 (16-bit form). The parameter "i" can be selected at address 10014.

c) Input interface program (14000-14036) - This program puts the ciphered text, already in 16-bit form, into 8-bit form to be handled by the encoding program. 8-bit characters are moved into memory locations 51000-52000.

d) Encoder program (14040-14152) - Encodes message and stores coded words into memory locations 52000-54000. Generator matrix is stored at

Memory location	Content
50200	104656
50202	46570
50204	23274
50206	11536

e) Noise generating program (14540-14754)

f) Noise mixing program (14756-15050) - Takes coded words from 52000-54000 and exclusive-ors them with noise words at 32000-34000, thus introducing noise into the text. Results are stored back at 52000-54000.

g) Minimum distance decoder (14154-14436) - Takes the distorted coded words from location 52000-54000, decodes them if they are correctible and stores the decoded words at location 56000-57000. Check polynomial is 11536 at address 50104.

h) Output interface program (14440-14464) - Takes decoded words and moves them to 30000-32000 to be deciphered.

i) Data-keyed deciphering program (10000-11044) - Same as (b), the only change needed is to change the contents of address 10012 from 40002 to 30002 to be compatible with the decipherment process. The program deciphers the message and stores the results in memory locations 40000-42000.

j) Output program (12000-12244) - Prints the cryptogram and the plaintext message.

VIII. SUMMARY AND CONCLUSIONS

After looking at the computer organization and establishing a basis to realize reversible transformations, three cryptographic systems were implemented:

1. Simple substitution
2. Pseudo-random cipher
3. Data-keyed cipher

The first, provided the basis for the other two. It was not intended to provide any significant amount of security since the cryptanalytic weakness of a simple substitution is well known.

The pseudo-random cipher is provided with a means to do polyalphabetic substitutions. This kind of cipher is known to be time consuming when done manually. The algorithm used to generate pseudo-random keys was a simple one, though it can be as complex as the user desires.

With the data-keyed cipher very significant results were obtained in the sense that its distribution plots were fairly flat. A disadvantage presented by this cipher was the error propagation when deciphering. This fact motivated the author to look into error correcting codes to use them with this or any other system. A (15,4) cyclic error correcting block code was implemented. This code contributed appreciably to reduce the probability of error,

P(e), when transmission was simulated over a noisy binary symmetric channel.

Finally, it can be said that the digital computer is suitable for encrypting and coding data for transmission, providing at the same time many different alternatives for both functions. With the advent of microprocessors and with communication systems tending to become all digital, it is certain that we will see in the future a computer performing these functions together with many more.

APPENDIX A. - PROGRAM FOR THE SIMPLE SUBSTITUTION CIPHER

```
010000 /005000
010002 /005002
010004 /005037
010006 /177560
010010 /105737
010012 /177560
010014 /100375
010016 /013700
010020 /177562
010022 /005003
010024 /020027
010026 /000260
010030 /100003
010032 /012703
010034 /000001
010036 /000416
010040 /020027
010042 /000300
010044 /100003
010046 /012703
010050 /000003
010052 /000410
010054 /020027
010056 /000320
010060 /100003
010062 /012703
010064 /000005
010066 /000402
010070 /012703
010072 /000007
010074 /005202
010076 /105737
010100 /177564
010102 /100375
010104 /110037
010106 /177566
010110 /005001
010112 /005037
010114 /177560
010116 /105737
010120 /177560
```

SIMPLE SUBSTITUTION PROGRAM... CONTINUATION

```
010122 /100375
010124 /013701
010126 /177562
010130 /122701
010132 /000215
010134 /001034
010136 /105737
010140 /177564
010142 /100375
010144 /110137
010146 /177566
010150 /012702
010152 /000012
010154 /105737
010156 /177564
010160 /100375
010162 /112737
010164 /000200
010166 /177566
010170 /077207
010172 /105737
010174 /177564
010176 /100375
010200 /112737
010202 /000212
010204 /177566
010206 /105737
010210 /177564
010212 /100375
010214 /112737
010216 /000212
010220 /177566
010222 /000137
010224 /001172
010226 /022703
010230 /000004
010232 /100455
010234 /022703
010236 /000002
010240 /100425
010242 /020127
010244 /000260
010246 /100003
010250 /012704
010252 /000260
```

SIMPLE SUBSTITUTION PROGRAM... CONTINUATION

```
010254 /000520
010256 /020127
010260 /000300
010262 /100003
010264 /012704
010266 /000260
010270 /000512
010272 /020127
010274 /000320
010276 /100003
010300 /012704
010302 /000260
010304 /000504
010306 /012704
010310 /000260
010312 /000501
010314 /020127
010316 /000260
010320 /100003
010322 /012704
010324 /000240
010326 /000473
010330 /020127
010332 /000300
010334 /100003
010336 /012704
010340 /000240
010342 /000465
010344 /020127
010346 /000320
010350 /100003
010352 /012704
010354 /000240
010356 /000457
010360 /012704
010362 /000240
010364 /000454
010366 /022703
010370 /000006
010372 /100425
010374 /020127
010376 /000260
010400 /100003
010402 /012704
010404 /000320
```

```
010406 /000443
010410 /020127
010412 /000300
010414 /100003
010416 /012704
010420 /000320
010422 /000435
010424 /020127
010426 /000320
010430 /100003
010432 /012704
010434 /000320
010436 /000427
010440 /012704
010442 /000320
010444 /000424
010446 /020127
010450 /000260
010452 /100003
010454 /012704
010456 /000300
010460 /000416
010462 /020127
010464 /000300
010466 /100003
010470 /012704
010472 /000300
010474 /000410
010476 /020127
010500 /000320
010502 /100003
010504 /012704
010506 /000300
010510 /000402
010512 /012704
010514 /000300
010516 /074001
010520 /074401
010522 /105737
010524 /177564
010526 /100375
010530 /110137
010532 /177566
010534 /005202
010536 /020227
```

SIMPLE SUBSTITUTION PROGRAM...CONTINUATION

```
010540 /000050
010542 /001036
010544 /005002
010546 /105737
010550 /177564
010552 /100375
010554 /112737
010556 /000215
010560 /177566
010562 /012702
010564 /000012
010566 /105737
010570 /177564
010572 /100375
010574 /112737
010576 /000200
010600 /177566
010602 /077207
010604 /105737
010606 /177564
010610 /100375
010612 /112737
010614 /000212
010616 /177566
010620 /105737
010622 /177564
010624 /100375
010626 /112737
010630 /000212
010632 /177566
010634 /005002
010636 /005004
010640 /000167
010642 /177244
```

APPENDIX B.- PROGRAM TO COUNT THE NUMBER OF OCCURRENCES OF EACH CHARACTER IN A MESSAGE STORED AT LOCATION 40000 AND UP

```
013000 /012704
013002 /117700
013004 /012702
013006 /000240
013010 /005003
013012 /012701
013014 /040000
013016 /021127
013020 /000215
013022 /001404
013024 /022102
013026 /001373
013030 /005203
013032 /000771
013034 /000240
013036 /000240
013040 /000240
013042 /105737
013044 /177564
013046 /100375
013050 /110237
013052 /177566
013054 /010324
013056 /005202
013060 /020227
013062 /000340
013064 /001351
013066 /000000
```

APPENDIX C - PROGRAM TO COMPUTE STATISTICS OF MESSAGE

```
10 BLKDEF B0,32,1
20 BLKDEF B1,32,0
30 BLKDEF B2,32,0
40 BLKDEF B3,32,1
50 LET B3,0,'@ABCDEFGHIJKLMNOPQRSTUVWXYZ[/]^_'
60 BIBSET B0,3,I1
65 BIBSET B0,1,15
66 BIBSET B3,1,15
70 LINK '110000',I1
150 FLOAT B0,B1
155 MOVE B1,B2
160 INTG B1
170 LET R0,B1,31
171 MOVE B2,B1
180 PRINT 'TOTAL NUMBER OF OCCURRENCES= ',R0
181 PRINT ' '
190 PRINT 'CHAR       NO. OF OCCURRENCES
200 FOR I2,0,31
210 LET R1,B1,I2
220 STACK 201,200,5,100.,4,254
240 LET B1,I2,R4
250 TRANS 0,B3,I2,I1
260 HOLOUT 'KB',I1,':'
270 LET R3,B1,I2
271 LET I3,B0,I2
280 PRINT '                 ',I3
282 NEXT I2
285 PRINT ' '
291 OSPEC 'CR'
292 DISPLY B1,'M','G'
293 OSPEC 'KB'
300 LET R1.32.
310 MOVE B1,B2
320 MUL B1,B1
330 INTG B2
340 LET R2,B2,31
350 QUOT R2,R2,R1
360 PRINT 'EXPECTED VALUE = ',R2
380 PROD R2,R2,R2
390 INTG B1
400 LET R3,B1,31
410 QUOT R3,R3,R1
420 DIF R3,R3,R2
430 PRINT 'VARIANCE = ',R3
450 STACK 203,16,255
460 PRINT 'STANDARD DEVIATION = ',R5
470 RETURN
END
```

APPENDIX D.- PROGRAM FOR THE

DATA-KEYED CIPHER

```
010000 /012737
010002 /040002
010004 /001006
010006 /012737
010010 /000007
010012 /001012
010014 /005037
010016 /037770
010020 /005000
010022 /005002
010024 /005037
010026 /177560
010030 /105737
010032 /177560
010034 /100375
010036 /013700
010040 /177562
010042 /005003
010044 /020027
010046 /000260
010050 /100003
010052 /012703
010054 /000001
010056 /000416
010060 /020027
010062 /000300
010064 /100003
010066 /012703
010070 /000003
010072 /000410
010074 /020027
010076 /000320
010100 /100003
010102 /012703
010104 /000005
010106 /000402
010110 /012703
010112 /000007
010114 /005202
010116 /105737
010120 /177564
```

DATA-KEYED PROGRAM...CONTINUATION

```
010122 /100375
010124 /110037
010126 /177566
010130 /010037
010132 /030000
010134 /010037
010136 /040000
010140 /012737
010142 /030002
010144 /001002
010146 /012737
010150 /040002
010152 /001004
010154 /005001
010156 /005037
010160 /177560
010162 /105737
010164 /177560
010166 /100375
010170 /013701
010172 /177562
010174 /013704
010176 /001002
010200 /010124
010202 /010437
010204 /001002
010206 /005004
010210 /022701
010212 /000215
010214 /001042
010216 /013704
010220 /001004
010222 /010114
010224 /105737
010226 /177564
010230 /100375
010232 /110137
010234 /177566
010236 /012702
010240 /000012
010242 /105737
010244 /177564
010246 /100375
010250 /112737
010252 /000200
```

DATA-KEYED PROGRAM... CONTINUATION

```
010254 /177566
010256 /077207
010260 /105737
010262 /177564
010264 /100375
010266 /112737
010270 /000212
010272 /177566
010274 /105737
010276 /177564
010300 /100375
010302 /112737
010304 /000212
010306 /177566
010310 /000137
010312 /001172
010314 /000240
010316 /022703
010320 /000004
010322 /100455
010324 /022703
010326 /000002
010330 /100425
010332 /020127
010334 /000260
010336 /100003
010340 /012704
010342 /000260
010344 /000520
010346 /020127
010350 /000300
010352 /100003
010354 /012704
010356 /000260
010360 /000512
010362 /020127
010364 /000320
010366 /100003
010370 /012704
010372 /000260
010374 /000504
010376 /012704
010400 /000260
010402 /000501
010404 /020127
```

DATA-KEYED PROGRAM...CONTINUATION

```
010406 /000260
010410 /100003
010412 /012704
010414 /000240
010416 /000473
010420 /020127
010422 /000300
010424 /100003
010426 /012704
010430 /000240
010432 /000465
010434 /020127
010436 /000320
010440 /100003
010442 /012704
010444 /000240
010446 /000457
010450 /012704
010452 /000240
010454 /000454
010456 /022703
010460 /000006
010462 /100425
010464 /020127
010466 /000260
010470 /100003
010472 /012704
010474 /000320
010476 /000443
010500 /020127
010502 /000300
010504 /100003
010506 /012704
010510 /000320
010512 /000435
010514 /020127
010516 /000320
010520 /100003
010522 /012704
010524 /000320
010526 /000427
010530 /012704
010532 /000320
010534 /000424
010536 /020127
```

DATA_KEYED PROGRAM...CONTINUATION

```
010540 /000260
010542 /100003
010544 /012704
010546 /000300
010550 /000416
010552 /020127
010554 /000300
010556 /100003
010560 /012704
010562 /000300
010564 /000410
010566 /020127
010570 /000320
010572 /100003
010574 /012704
010576 /000300
010600 /000402
010602 /012704
010604 /000300
010606 /074001
010610 /074401
010612 /023737
010614 /001012
010616 /037770
010620 /100024
010622 /013704
010624 /001006
010626 /012437
010630 /001014
010632 /010437
010634 /001006
010636 /012704
010640 /000004
010642 /106337
010644 /001014
010646 /077403
010650 /000241
010652 /012704
010654 /000005
010656 /106137
010660 /001014
010662 /077403
010664 /013704
010666 /001014
010670 /074401
```

DATA-KEYED PROGRAM...CONTINUATION

```
010672 /005004
010674 /000240
010676 /000240
010700 /000240
010702 /105737
010704 /177564
010706 /100375
010710 /110137
010712 /177566
010714 /013704
010716 /001004
010720 /010124
010722 /010437
010724 /001004
010726 /005237
010730 /037770
010732 /005202
010734 /020227
010736 /000050
010740 /001036
010742 /005002
010744 /105737
010746 /177564
010750 /100375
010752 /112737
010754 /000215
010756 /177566
010760 /012702
010762 /000012
010764 /105737
010766 /177564
010770 /100375
010772 /112737
010774 /000200
010776 /177566
011000 /077207
011002 /105737
011004 /177564
011006 /100375
011010 /112737
011012 /000212
011014 /177566
011016 /105737
011020 /177564
011022 /100375
```

DATA-KEYED PROGRAM... CONTINUATION

```
011024 /112737
011026 /000212
011030 /177566
011032 /005002
011034 /005004
011036 /000167
011040 /177112
```

APPENDIX E. - ENCODING PROGRAM FOR THE (15,4) CYCLIC CODE

```
014040 /012700
014042 /051000
014044 /000240
014046 /000240
014050 /013702
014052 /050100
014054 /112037
014056 /050140
014060 /012703
014062 /000002
014064 /012704
014066 /000004
014070 /012705
014072 /050200
014074 /005037
014076 /050142
014100 /012501
014102 /106337
014104 /050140
014106 /103002
014110 /074137
014112 /050142
014114 /000240
014116 /077410
014120 /013737
014122 /050142
014124 /052000
014126 /005237
014130 /014124
014132 /005237
014134 /014124
014136 /077326
014140 /077233
014142 /012737
014144 /052000
014146 /020210
014150 /000137
014152 /001172
```

APPENDIX F. - NOISE GENERATING PROGRAM

```
014540 /012700
014542 /032000
014544 /012701
014546 /001000
014550 /005020
014552 /077102
014554 /000240
014556 /012700
014560 /057000
014562 /012746
014564 /012705
014566 /012746
014570 /000030
014572 /011667
014574 /000026
014576 /012704
014600 /177304
014602 /012714
014604 /010000
014606 /012637
014610 /177300
014612 /011467
014614 /000030
014616 /012701
014620 /177316
014622 /012703
014624 /000030
014626 /012624
014630 /012714
014632 /000401
014634 /014446
014636 /062716
014640 /000003
014642 /077307
014644 /005327
014646 /000000
014650 /001414
014652 /011614
014654 /005044
014656 /012711
014660 /177775
014662 /005724
014664 /042714
014666 /000001
014670 /060014
```

NOISE GENERATING PROGRAM...CONTINUATION

```
014672 /012774
014674 /000001
014676 /000000
014700 /000750
014702 /005026
014704 /012700
014706 /057000
014710 /012701
014712 /032000
014714 /012702
014716 /000177
014720 /012703
014722 /000020
014724 /006220
014726 /006011
014730 /077303
014732 /005721
014734 /012703
014736 /000005
014740 /006220
014742 /006011
014744 /077303
014746 /005721
014750 /077215
014752 /000137
014754 /001172
```

APPENDIX G - DECODING PROGRAM FOR THE MINIMUM DISTANCE DECODER

```
014154 /012700
014156 /052000
014160 /013737
014162 /050100
014164 /050102
014166 /063737
014170 /050100
014172 /050102
014174 /013701
014176 /050104
014200 /012703
014202 /054000
014204 /012704
014206 /000017
014210 /005037
014212 /050116
014214 /011005
014216 /074105
014220 /012702
014222 /000017
014224 /006305
014226 /005537
014230 /050116
014232 /077204
014234 /022737
014236 /000004
014240 /050116
014242 /002010
014244 /006301
014246 /103402
014250 /077421
014252 /000407
014254 /062701
014256 /000002
014260 /077425
014262 /000403
014264 /010123
014266 /005720
014270 /000403
014272 /012723
014274 /000000
*
```

```
..
014276 /005720
014300 /162737
014302 /000001
014304 /050102
014306 /003336
014310 /000240
014312 /000240
014314 /000240
014316 /000240
014320 /013700
014322 /050100
014324 /012701
014326 /054001
014330 /012702
014332 /056000
014334 /005003
014336 /005004
014340 /112103
014342 /005201
014344 /112104
014346 /005201
014350 /012705
014352 /000005
014354 /000241
014356 /106103
014360 /077502
014362 /012705
014364 /000004
014366 /106303
014370 /077502
014372 /012705
014374 /000005
014376 /000241
014400 /106104
014402 /077502
014404 /012705
014406 /000004
014410 /106304
014412 /077502
014414 /012705
014416 /000005
014420 /000241
014422 /106104
014424 /077502
014426 /074304
014430 /110422
014432 /077040
014434 /000137
014436 /001172
```

LIST OF REFERENCES

1. Westing, A., Privacy and Freedom, Atheneum 1967.

2. Savage, J.E., "Some Simple Self-Synchronizing Digital Data Scramblers," Bell System Technical Journal, Vol. 45, No. 2, February 1967.

3. Leeper, D.G., "A Universal Digital Data Scrambler," Bell System Technical Journal, Vol. 52, No. 10, December, 1973.

4. Gitlin, R.D. and Hayes, J.F., "Timing Recovery and Scramblers in Data Transmission," Bell System Technical Journal, Vol. 54, No. 3, March 1975.

5. Mellen, G.E., "Cryptology, Computers and Common Sense," 1973 FJCC, AFIPS Conference.

6. Twigg, T., "Need To Keep Digital Data Secure?" Electronic Design, Vol. 23, No. 68, Pg. 68-76, 1972.

7. Henricksson, V., "On A Scrambling Property of Feedback Shift Registers," IEEE Transactions on Communications, Vol. 20, No. 5, Pp. 998-1001, October 1972.

8. Golob, S.W., Shift Register Sequences, San Francisco: Holden-Day 1967.

9. Meyer, C.H. and Tochman, W.L., "Pseudorandom Codes Can Be Cracked," Electronic Design, Vol. 23, No. 74, Pp. 74-76, 1972.

10. Naval Research Laboratory Report 7900, "Cryptographic Digital Communications," by Torrieri, D.J., July 1975.

11. Geffe, P.R., "Secure Electronic Cryptography", Westinghouse Electric Corporation, Baltimore, Md., Pp. 181-187, 1972.

12. Altman, L., Microprocessors, Electronics Book Series, McGraw-Hill, 1975.

13. Kahn, D., The Codebreakers, Macmillan, New York, 1967.

14. Shannon, C.E. and Weaver, W., The Mathematical Theory of Communications, University of Illinois Press, Urbana, Ill., 1949.

15. Shannon, C.E., "Communication Theory of Secrecy Systems," Bell System Technical Journal, 28,656, 1949.

16. Vernam, G.S., "Cipher Printing Telegraph Systems," Journal of the AIEF, Vol. XLV, February, 1926.

17. Sinkov, A., Elementary Cryptanalysis: A Mathematical Approach, Random House, New York, 1968.

18. Ash, Robert B., Information Theory.

19. S.G.S. Shiva, "Some Results on Binary Codes with Equivalent Words," IEEE Transaction on Information Theory, March 1969, Volume IT-15, Number 2.

20. Peterson, W. and Weldon, E.J. Jr., Error Correcting Codes.

21. Centinyilmaz, N., Application of the Computer for Real Time Encoding and Decoding of Cyclic Block Codes, Master's Thesis, Naval Postgraduate School, December 1975.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0212 Naval Postgraduate School Monterey, California 93940	2
3.	Department Chairman, Code 62 Department of Electrical Engineering Naval Postgraduate School Monterey, California 93940	1
4.	Professor George H. Marmont Code 62Ma (Thesis Advisor) Naval Postgraduate School Monterey, California 93940	1
5.	LtCol. Robert W. Burton, USAF Code 62Zn (Second Reader) Naval Postgraduate School Monterey, California 93940	1
6.	LT. Eduardo E. Coquis R. Dirección de Instrucción Ministerio de Marina Lima PERU	2
7.	Dirección de Instrucción Ministerio de Marina Lima PERU	1